Procédures de décision par élicitation incrémentale de préférences en optimisation multicritère, multi-agents et dans l'incertain

Nawal Benabbou

Université Pierre et Marie Curie (UFR d'Ingénierie)
École Doctorale Informatique, Télécommunications et Électronique
Laboratoire d'Informatique de Paris 6 (Équipe DECISION)

THÈSE DE DOCTORAT EN INFORMATIQUE

Spécialité : Aide à la décision

Procédures de décision par élicitation incrémentale de préférences en optimisation multicritère, multi-agents et dans l'incertain

---

Présentée et soutenue publiquement le 5 mai 2017 par Nawal Benabbou

JURY :

Directeur de thèse	Patrice PERNY Professeur à l'Université Pierre et Marie Curie
Rapporteurs	Jérôme LANG Directeur de Recherche CNRS à l'Université Paris-Dauphine Marc PIRLOT Professeur à la Faculté Polytechnique de Mons
Examinateurs	Clarisse DHAENENS Professeur à l'Université Lille 1 Nicolas MAUDET Professeur à l'Université Pierre et Marie Curie Vincent MOUSSEAU Professeur à l'École Centrale de Paris Daniel VANDERPOOTEN Professeur à l'Université Paris-Dauphine

À ma mère,
la reine de mon cœur.

Remerciements

En premier lieu, je tiens à exprimer ma plus profonde gratitude à Patrice Perny, Professeur à l'Université Pierre et Marie Curie, qui a pleinement encadré mes recherches dans le cadre du doctorat. Je le remercie vivement pour sa motivation, sa disponibilité, son enthousiasme, sa gentillesse, ses conseils, ses encouragements ainsi que pour son investissement personnel qui témoigne de son engagement à l'égard de l'avenir professionnel de ses doctorants. Outre ses qualités humaines qui ont rendu le travail à ses côtés très agréable, sa passion pour l'excellence a été une véritable source d'inspiration pour moi durant ces années de thèse. J'encourage ainsi tous les futurs doctorants dans le domaine de la théorie de la décision algorithmique à le choisir comme directeur de thèse pour avoir le privilège de suivre son excellente formation au métier d'enseignant-chercheur.

Je tiens par ailleurs à remercier Jérôme Lang, Directeur de recherche CNRS à l'Université Paris-Dauphine, et Marc Pirlot, Professeur à la Faculté Polytechnique de Mons, pour l'intérêt qu'ils ont porté à mes travaux de recherche en me faisant l'honneur d'accepter la charge de rapporter cette thèse. Je remercie également Clarisse Dhaenens, Professeur à l'Université Lille 1, Nicolas Maudet, Professeur à l'Université Pierre et Marie Curie, Vincent Mousseau, Professeur à l'École Centrale de Paris et Daniel Vanderpooten, Professeur à l'Université Paris-Dauphine, pour s'être intéressés à mes travaux de recherche et pour avoir accepté de participer à l'évaluation de cette thèse.

Je tiens également à remercier chaleureusement tous les membres des équipes DECISION et RO du LIP6, sans qui ces années de doctorat n'auraient pas été aussi agréables. Je remercie en particulier Paolo Viappiani pour ses nombreux conseils, Olivier Spanjaard pour son soutien et sa gentillesse, Thibaut Lust pour les innombrables fous rires partagés, Safia Kedad-Sidhoum pour son enthousiasme naturel et sa bonne humeur communicative, Pierre Fouilloux pour toutes ces discussions riches d'enseignement, Bruno Escoffier pour son humour intarissable à la pause déjeuner, Fanny Pascual pour sa bienveillance, Emmanuel Hyon pour ses visites surprises très sympathiques, Christoph Dürr pour tous ses petits goûters organisés dans la salle de repos, ainsi que Evripidis Bampis, Christophe Gonzales, Viet Hung Nguyen, Spyros Angelopoulos, Paul Weng et Pierre-Henri Wuillemin, avec qui j'ai également eu le plaisir de discuter de temps en temps.

Je tiens aussi à remercier tous les doctorants et chercheurs postdoctoraux des équipes DECISION et RO du LIP6, avec une mention particulière pour Siao-Leu Phouratsamay pour son soutien et sa personnalité attachante, Cécile Rottner pour ses ondes positives et son sourire contagieux, Hamza Agli pour son amitié et ses encouragements, Hugo Gilbert pour toutes ses anecdotes cocasses et sympathiques,

Angelina Vidali pour ses nombreux conseils, Ahmed Mabrouk pour sa grande sympathie et Morgan Chopin pour sa bonne humeur permanente.

Ce passage de remerciements serait bien incomplet si je ne remerciais pas également ma famille (ma mère Samia Djamila, mes sœurs Sihem, Linda et Kahina, mes beaux-frères Julien et Nasser, mes neveux Ismael, Sofia, Djeena, Issa, Djibril, Idriss, Ahmada, etc.) qui constitue une source de bonheur et d'énergie inépuisable au quotidien. Enfin, je tiens à remercier mon conjoint Samuel pour avoir su m'apporter un soutien indéfectible durant toutes mes études universitaires, pour avoir pris soin de moi pendant ces années de thèse et pour m'avoir offert tout son amour qui contribue quotidiennement à mon épanouissement personnel et professionnel.

Table des matières

Introduction

La théorie de la décision est un domaine de recherche au carrefour de multiples disciplines, comme l'économie, la sociologie, la philosophie, la psychologie ou encore les sciences de gestion, et qui occupe actuellement une place importante en informatique. Pour cette dernière discipline, il peut s'agir de mettre en œuvre des systèmes autonomes, prenant des décisions de façon automatique, en tenant compte des préférences des agents exploitant ces systèmes (pilotage automatique de processus, véhicules autonomes, robots intelligents...). Les outils informatiques permettent par ailleurs de concevoir des systèmes d'aide à la décision dont la fonction est d'assister la décision humaine en facilitant l'exploration des différentes alternatives du problème. Citons par exemple les systèmes de recommandation que l'on trouve sur le web pour l'achat d'un bien (produit hi-tech, véhicule automobile, vêtement...), le choix d'un séjour de vacances ou encore la sélection de films/séries dans une base de données de vidéos. Ces systèmes servent également dans des contextes où les enjeux de la décision sont plus importants, qu'il s'agisse de l'achat d'un appartement, de la planification de traitements thérapeutiques, de la gestion de politiques publiques ou encore de choix stratégiques pour une organisation (allocation de ressources, optimisation de réseaux de communication, problèmes de transport, stratégie marketing...).

Dans la littérature, différents modèles décisionnels ont été proposés pour représenter de manière efficace les préférences d'un agent dans des situations de décision multicritère, multi-agents et/ou dans l'incertain. Il s'agit d'outils formels étudiés sur la base de critères prescriptifs (capacité de recommandation), descriptifs (aptitude à simuler des comportements observés) ou encore normatifs (comportement optimal en théorie) permettant de caractériser les meilleures alternatives du problème. La coexistence de plusieurs modèles décisionnels s'explique notamment par le fait que la notion d'optimalité ne va plus de soi dès lors que les alternatives sont évaluées selon différents points de vue (critères, agents et/ou scénarios), souvent conflictuels par nature (comme le prix et la qualité d'un produit). Cette difficulté a conduit les chercheurs en théorie de la décision à proposer différents modèles décisionnels paramétriques, plus ou moins sophistiqués, permettant de représenter les préférences d'un individu dans des contextes de décision multi-objectifs. Ces modèles décisionnels permettent de se ramener au cas simple d'alternatives mono-évaluées en attribuant à chaque décision possible une valeur unique, obtenue par agrégation de ses différentes évaluations (performances sur les différents critères ou dans les différents scénarios). Parmi ces modèles, nous pouvons mentionner la somme pondérée, la norme de Tchebycheff [Wierzbicki, 1986], l'opérateur OWA [Yager, 1988], l'agrégateur WOWA [Torra, 1997], l'intégrale de Cho-

quet [Grabisch and Labreuche, 2010], le modèle EU [von Neumann and Morgenstern, 1947], le modèle RDU [Quiggin, 1992] ou encore le modèle CEU [Schmeidler, 1986]. Lorsque les préférences de plusieurs agents sont à prendre en considération conjointement, il s'agit ensuite de combiner les préférences individuelles pour pouvoir déterminer la meilleure décision du point de vue collectif.

Si ces modèles décisionnels caractérisent toutes les alternatives optimales, ces derniers ne donnent pas automatiquement les moyens de les calculer efficacement dans des situations où les alternatives sont très nombreuses et définies de manière implicite par un problème d'optimisation combinatoire. Ces préoccupations algorithmiques liées à la représentation des préférences relèvent de la théorie de la décision algorithmique, une thématique récente qui se développe rapidement au carrefour de la théorie de la décision, de la recherche opérationnelle et de l'intelligence artificielle. Pour de nombreux modèles décisionnels, des algorithmes ont été conçus pour la détermination efficace des alternatives préférées sur domaine combinatoire. Par exemple, nous pouvons citer des solutions algorithmiques proposées pour des fonctions d'utilité non additives (e.g., [Tsaggouris and Zaroliagis, 2009b]), pour les opérateurs OWA (e.g., [Ogryczak and Śliwiński, 2003, Galand and Spanjaard, 2012]), pour les normes de Tchebycheff (e.g., [Clímaco et al., 2006, Galand and Perny, 2006]), pour les intégrales de Choquet (e.g., [Galand and Perny, 2007, Galand et al., 2013]), pour les opérateurs WOWA (e.g., [Ogryczak and Śliwiński, 2009]) ainsi que pour le modèle RDU (e.g., [Nielsen and Jaffray, 2006, Jeantet et al., 2012]). Cependant, ces algorithmes nécessitent de connaître les paramètres du modèle (jeu de poids, capacité, fonctions d'utilité...) permettant de rendre compte des préférences des agents, ce qui est généralement difficile à réaliser de manière précise en pratique. Ainsi, outre la difficulté d'explorer des espaces de solutions de grande taille, l'indétermination sur les paramètres constitue une source de complexité supplémentaire dans l'analyse de problèmes de décision multi-objectifs sur domaine combinatoire.

L'élicitation des préférences pour l'aide à la décision est une problématique de recherche dont l'objectif est de concevoir des méthodes permettant de formuler des recommandations personnalisées en recueillant des informations sur les préférences des agents. Pour obtenir ces informations, ces méthodes doivent généralement solliciter directement les agents en leur posant des questions sur leurs préférences subjectives. De ce fait, la quantité de données que ces méthodes sont en mesure de collecter en pratique est relativement faible. L'élicitation des préférences avec information limitée est reconnue comme étant une tâche cruciale dans de nombreux domaines d'application, comme les systèmes de recommandation ou les assistants personnels (e.g., [Peintner et al., 2008, Boutilier, 2013]). Dans l'optique de limiter le nombre de questions nécessaires à la prise de décision, des procédures d'élicitation partielle ont été proposées : il s'agit de poser des questions aux agents, soigneusement choisies les unes après les autres, pour réduire progressivement l'imprécision sur les paramètres du modèle, de manière à formuler rapidement une recommandation (e.g., [White III et al., 1984, Wang and Boutilier, 2003]). Cette approche dite "incrémentale" s'est montrée particulièrement efficace sur une étude menée avec de vrais utilisateurs [Braziunas and Boutilier, 2010]. C'est pourquoi, dans le cadre de cette thèse, nous nous attelons à étendre efficacement cette approche non seulement aux cas de modèles décisionnels complexes, mais aussi aux situations de décision

sur domaine combinatoire. En effet, l'élicitation des préférences sur domaine combinatoire représente un challenge supplémentaire à relever car il s'agit de parvenir à discriminer entre toutes les alternatives possibles, aussi nombreuses soient-elles. Cette difficulté a motivé des contributions dans diverses domaines, comme par exemple dans les espaces multi-attributs [Gonzales and Perny, 2004, Braziunas and Boutilier, 2007, Koriche and Zanuttini, 2010], les problèmes de mariages stables [Drummond and Boutilier, 2014], les problèmes de satisfaction de contraintes [Boutilier et al., 2006], les processus décisionnels Markoviens [Regan and Boutilier, 2011, Weng and Zanuttini, 2013, Gilbert et al., 2015] ou encore en optimisation multi-objectifs [Vanderpooten and Vincke, 1997, Korhonen, 2005, Kaddani et al., 2017].

Positionnement de la thèse. Dans le cadre de cette thèse, nous proposons et étudions une nouvelle approche combinant l'élicitation incrémentale et l'exploration implicite des solutions potentielles, permettant de réduire l'imprécision sur les paramètres du modèle décisionnel en cours de résolution, pour pouvoir centrer rapidement la recherche sur les solutions préférées. L'idée est de poser des questions à des moments clés de la résolution pour collecter uniquement des informations servant à discriminer entre les alternatives du problème. La classe de problèmes concernés inclut une large variété de problèmes d'optimisation concrets qui se formalisent comme des problèmes de graphes ou de programmation linéaire en nombres entiers (plus court chemin, arbre couvrant de poids minimum, affectation, sac à dos...).

Organisation du document. Cette thèse se décompose en cinq chapitres. Le premier chapitre propose une introduction à la modélisation des préférences en décision multicritère, multi-agents et dans le risque, avant de faire le tour des approches classiques pour la problématique d'élicitation des préférences. Le deuxième chapitre expose nos travaux sur l'élicitation incrémentale d'une intégrale de Choquet dans le cadre des trois grandes problématiques étudiées en aide à la décision : le choix, le rangement et le tri [Roy, 1996]. Le troisième chapitre porte sur la décision par élicitation incrémentale en optimisation combinatoire multicritère. Plus précisément, nous proposons des algorithmes combinant résolution et élicitation incrémentale pour la problématique de recherche dans un graphe d'états multicritère ainsi que pour le problème d'arbres couvrants multicritères. Le quatrième chapitre se concentre sur des problèmes de décision collective en présence de préférences imprécisément connues. Dans ce chapitre, nous proposons une méthode d'élicitation incrémentale pour la méthode de Borda avec un ensemble d'alternatives donné en extension. Nous introduisons aussi un algorithme de recherche par élicitation incrémentale pour résoudre le problème du sac à dos multi-agents en utilisant le vote par approbation. Enfin, le cinquième chapitre concerne la décision séquentielle dans le risque avec des utilités imprécisément définies. Dans ce cadre, nous proposons un algorithme de recherche dans un arbre de décision exploitant l'élicitation incrémentale pour construire rapidement une stratégie nécessairement optimale avec le modèle EU [von Neumann and Morgenstern, 1947].

Chapitre 1

Modélisation et élicitation des préférences

Résumé

Ce chapitre a pour objectif de présenter les modèles décisionnels qui ont été étudiés durant cette thèse ainsi que les approches classiques pour l'élicitation des préférences. La première section s'intéresse à la modélisation des préférences dans une situation de décision multicritère. Il s'agit de comparer les différents modèles décisionnels sur la base de propriétés mathématiques désirées et sur leur capacité à reproduire des comportements observés. Nous nous intéresserons plus particulièrement aux sommes pondérées, aux opérateurs OWA et aux intégrales de Choquet. La seconde section étudie la problématique de décision collective sous l'angle de la théorie du vote. Après une discussion sur la pertinence de la règle de vote majoritaire, nous présenterons les difficultés sous-jacentes de l'agrégation des préférences de manière plus générale. En particulier, nous présenterons le théorème d'impossibilité d'Arrow ainsi que celui de Gibbard et Satterthwaite. Nous situerons ensuite la méthode de Borda et le vote par approbation par rapport à ces résultats d'impossibilité. La troisième section est dédiée à la problématique de décision dans le risque. Nous commençons par y introduire les dominances stochastiques en discutant leurs interprétations en terme de préférences. Nous présenterons ensuite la théorie de l'utilité espérée pour la modélisation des préférences dans le risque. Enfin, la dernière section de ce chapitre est consacrée à la problématique d'élicitation des préférences. Nous commencerons par présenter un éventail des approches standards, avant de décrire en détails l'élicitation incrémentale fondée sur le Minimax Regret.

Introduction

La prise de décision est un acte omniprésent dans le quotidien de chacun. Il faut par exemple décider du chemin à emprunter pour aller au travail ou encore de sa tenue pour la journée. Prendre une décision peut être une tâche difficile dans certaines situations, notamment lorsque les alternatives possibles sont en très grand nombre, que les avis de plusieurs agents sont à prendre en considération, que les conséquences de chaque décision sont imprécisément connues, ou encore que les alternatives sont évaluées selon plusieurs critères conflictuels (comme la qualité et le prix). Quand le problème de décision représente un enjeu important (comme le choix d'une stratégie commerciale), il est primordial pour celui ou ceux assumant la responsabilité de l'acte décisionnel de bénéficier d'une assistance permettant de déterminer la décision la plus adaptée ; cet individu ou groupe d'individus est communément appelé *décideur* dans la littérature.

Cette assistance est généralement apportée par un analyste spécialisé dans le domaine de la théorie de la décision qui est notamment en charge d'analyser la situation décisionnelle. Il s'agit pour l'analyste de formaliser le problème de décision, de définir les alternatives possibles et leurs conséquences respectives, d'élaborer des critères d'évaluation, ou encore de collecter des informations sur les préférences subjectives du décideur. Cette étape, que nous appelons *phase de structuration*, requiert une attention particulière de la part de l'analyste car celle-ci résume toutes les informations à prendre en compte pour parvenir à déterminer la décision la plus appropriée ; en réalité, la structuration de problème constitue le principal sujet de la théorie de la décision (e.g., [von Winterfeldt, 1980, Rosenhead and Mingers, 2001, Mingers and Rosenhead, 2004, Greco et al., 2010a]).

La phase de structuration est en pratique relativement complexe à mettre en œuvre, le décideur rencontrant très souvent des difficultés à exprimer ce qu'il recherche précisément. En particulier, la théorie de la décision a révélé et étudié trois causes possibles à l'origine de ces difficultés :

- la présence de plusieurs critères potentiellement conflictuels à prendre en compte simultanément,
- l'existence d'opinions divergentes dans le cadre de problèmes de décision collective,
- et l'impossibilité de prévoir avec certitude les conséquences des décisions possibles.

Ces trois sources de complexité sont respectivement à l'origine de la théorie de la décision multicritère, de la théorie du choix social et de la théorie de la décision dans l'incertain.

Après l'étape de structuration vient ce que nous appelons la *phase de résolution* consistant essentiellement à tenir compte de toutes les informations provenant de la phase de structuration afin de formuler une recommandation pertinente au décideur. Le type de recommandation dépend de la problématique considérée. Les trois grandes problématiques étudiées en aide à la décision sont les suivantes [Roy, 1996] :

- le choix, dont l'objectif est de déterminer l'alternative que le décideur préfère,
- le rangement, consistant à ordonner toutes les alternatives par ordre de préférence,
- le tri, dont le but est de répartir toutes les solutions dans des catégories prédéfinies, en se basant sur une évaluation intrinsèque des alternatives, ainsi que sur les préférences du décideur.

Cette phase de résolution peut se révéler particulièrement difficile lorsque les préférences du décideur sont complexes ou que les alternatives possibles sont très nombreuses et définies de manière implicite (comme dans les problèmes de satisfaction de contraintes et de planification de tâches). Les situations

décisionnelles de ce type ont participé à l'émergence de la théorie de la décision algorithmique, domaine mêlant préoccupations algorithmiques et théorie de la décision. Par exemple, l'optimisation combinatoire multi-objectifs fait partie des principales problématiques étudiées dans ce domaine (e.g., [Ehrgott, 2006, Greco et al., 2016]). Dans ce chapitre, nous commençons par étudier la problématique de modélisation des préférences en décision multicritère (cf. Section 1.1), multi-agents (cf. Section 1.2) et dans l'incertain (cf. Section 1.3), puis nous nous intéresserons aux différentes approches proposées dans la littérature pour l'élicitation des préférences.

1.1 Décision multicritère

Un problème de décision multicritère se caractérise par la prise en compte explicite de plusieurs critères qui traduisent les différents objectifs du décideur, chaque critère servant à mesurer les préférences du décideur selon un certain point de vue comme la qualité, le prix ou encore la durée. Un critère est formellement défini par une fonction g_j dont le domaine de définition est l'ensemble des décisions possibles et dont les images sont prises dans un ensemble X_j totalement ordonné (e.g., [Vincke, 1989]) ; nous ferons souvent l'hypothèse que $X_j \subseteq \mathbb{R}$ pour tout $j \in \mathcal{Q} = \{1, \ldots, q\}$ où q représente le nombre de critères du problème. L'ensemble $X = X_1 \times \ldots \times X_q$, communément appelé *espace des critères*, constitue un nouvel espace de description des décisions possibles. Plus précisément, chaque décision x est décrite par le vecteur $(g_1(x), \ldots, g_q(x)) \in X$, où $g_j(x)$ représente la performance (aussi appelée *utilité*) de la décision x par rapport au $j^{\text{ème}}$ point de vue. Dans la suite, nous noterons $\mathcal{X}$ l'image des décisions possibles dans l'espace des critères ($\mathcal{X} \subset X$ en général). Par ailleurs, nous poserons $x_j = g_j(x)$ pour tout $j \in \mathcal{Q}$ pour simplifier les notations. Enfin, sans perte de généralité, nous supposerons que les décisions possibles x sont d'autant meilleures que leurs performances $x_j, j \in \mathcal{Q}$, sont élevées. En effet, nous pouvons toujours nous ramener à ce cas moyennant un changement de signe.

Chaque critère g_j induit un préordre total sur $\mathcal{X}$, noté $\succsim_j$, représentant les préférences du décideur vis-à-vis du $j^{\text{ème}}$ point de vue. Plus précisément, la relation de préférence $\succsim_j$ est définie par :

$$\forall x, y \in \mathcal{X},\ x \succsim_j y \Leftrightarrow x_j \geq y_j$$

où $x \succsim_j y$ signifie que le vecteur x est au moins aussi bon que le vecteur y selon le $j^{\text{ème}}$ point de vue. Bien que les relations $\succsim_j, j \in \mathcal{Q}$, permettent de comparer toutes les décisions possibles par rapport à chaque critère, celles-ci sont généralement insuffisantes pour résoudre le problème de décision dont il est question. En effet, la principale difficulté réside dans le fait que les critères considérés sont très souvent conflictuels : améliorer une performance réalisée sur un critère conduit très souvent à la détérioration de celle obtenue sur un autre critère. En particulier, aucune décision n'est meilleure que toutes les autres sur tous les critères à la fois. Cette difficulté a donné lieu à de nombreuses contributions sur la modélisation des préférences dans un contexte multicritère. Dans cette section, nous nous intéressons à cette problématique en commençant par présenter la dominance de Pareto, qui est une règle de dominance très naturelle dans les situations où plusieurs critères sont à optimiser simultanément. Nous nous intéresserons ensuite à la

modélisation des préférences par agrégation des critères. Nous exposerons notamment quelques propriétés mathématiques qu'il convient d'imposer à la fonction d'agrégation pour pouvoir modéliser au mieux les préférences du décideur. Enfin, nous terminerons cette section par une présentation des fonctions d'agrégation que nous avons étudiées durant cette thèse. Sauf mention contraire, nous nous plaçons dans le cadre de la problématique du choix tout au long de cette section : il s'agit de déterminer la meilleure décision possible pour le décideur. Par ailleurs, nous utiliserons le terme *alternative* pour désigner aussi bien une décision possible que le vecteur de performances qui lui est associé.

1.1.1 Dominance de Pareto et agrégation multicritère

Avant de s'intéresser aux préférences subjectives du décideur, il est généralement possible d'éliminer du problème des alternatives qui, par comparaison avec d'autres, sont indéniablement sous-optimales. Par exemple, si une alternative est meilleure qu'une autre sur tous les critères à la fois, alors nous pouvons penser que la première alternative sera préférée à la seconde par tout individu "rationnel". Cette observation a donné naissance à la relation de dominance suivante :

Définition 1 (Pareto-dominance faible)**.** *Une alternative $x \in \mathcal{X}$ est dominée faiblement au sens de Pareto par une alternative $y \in \mathcal{X}$, noté $y \succsim_P x$, si et seulement si : $\forall j \in \mathcal{Q},\ y_j \geq x_j$.*

En d'autres termes, une alternative x est dominée faiblement au sens de Pareto par une alternative y si et seulement si y est au moins aussi bonne que x sur chaque point de vue considéré. Dans ce cas, nous pouvons aussi dire que y domine faiblement x au sens de Pareto. Notons que deux alternatives ayant la même évaluation vectorielle se dominent l'une l'autre avec cette relation de dominance. Dans la littérature, il existe des renforcements de cette relation, exigeant que y soit strictement meilleure que x sur au moins un critère :

Définition 2 (Pareto-dominance)**.** *Une alternative $x \in \mathcal{X}$ est dominée au sens de Pareto par une alternative $y \in \mathcal{X}$, noté $y \succ_P x$, si et seulement si : $\forall j \in \mathcal{Q},\ y_j \geq x_j$ et $\exists j \in \mathcal{Q},\ y_j > x_j$.*

Définition 3 (Pareto-dominance stricte)**.** *Une alternative $x \in \mathcal{X}$ est dominée strictement au sens de Pareto par une alternative $y \in \mathcal{X}$, noté $y \gg_P x$, si et seulement si : $\forall j \in \mathcal{Q},\ y_j > x_j$.*

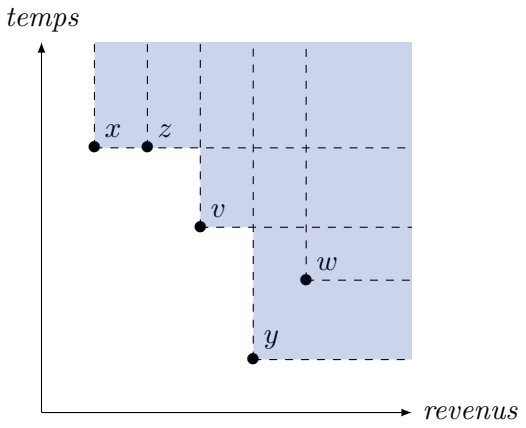
Remarquons que ces trois relations de dominance sont liées : la Pareto-dominance stricte implique la Pareto-dominance, cette dernière induisant la Pareto-dominance faible. Ces relations de dominance produisent des ordres partiels sur les alternatives qui permettent d'éliminer a priori des alternatives non pertinentes du problème. Ces relations conduisent en particulier aux notions d'optimalité suivantes :

Définition 4 (Pareto-optimalité)**.** *L'ensemble* $\mathrm{ND}(\mathcal{X})$ *des alternatives non dominées au sens de la dominance de Pareto est défini par :* $\mathrm{ND}(\mathcal{X}) = \{x \in \mathcal{X} : \forall y \in \mathcal{X}, \neg(y \succ_P x)\}$.

Définition 5 (Pareto-optimalité faible)**.** *L'ensemble* $\mathrm{SND}(\mathcal{X})$ *des alternatives non dominées au sens de la dominance stricte de Pareto est défini par :* $\mathrm{SND}(\mathcal{X}) = \{x \in \mathcal{X} : \forall y \in \mathcal{X}, \neg(y \gg_P x)\}$.

où $\neg$ représente l'opérateur de négation logique. Les décisions associées aux éléments de $\mathrm{ND}(\mathcal{X})$ sont communément appelées solutions Pareto-optimales, tandis que leur image dans l'espace des critères est dite Pareto-efficace. La détermination de l'ensemble $\mathrm{ND}(\mathcal{X})$ fait partie des sujets phares en optimisation combinatoire multi-objectifs (e.g., [Ehrgott, 2006, Greco et al., 2016]). Par ailleurs, les décisions correspondant aux vecteurs de $\mathrm{SND}(\mathcal{X})$ sont souvent appelées solutions faiblement Pareto-optimales, leur image dans l'espace des critères étant dite faiblement Pareto-efficace. L'exemple suivant permet d'illustrer graphiquement ces deux notions d'optimalité :

Exemple 1. *Un individu souhaite comparer différentes possibilités d'évolution de carrière pour pouvoir augmenter ses revenus (premier critère) tout en préservant suffisamment de temps libre (deuxième critère) afin de pouvoir concilier vie privée et vie professionnelle. Les cinq alternatives possibles sont représentées ci-dessous dans l'espace des critères :*



Sur ce graphique, chaque point correspond au vecteur de performances d'une alternative du problème. Le cône de dominance *d'un point est la zone délimitée par les deux traits pointillés sortant de ce point. Ce cône contient toutes les alternatives dominant ce point au sens de Pareto ; en excluant les bords du cône, nous obtenons toutes celles qui le dominent strictement. Graphiquement, nous voyons que les alternatives v, w et z sont Pareto-optimales car aucune alternative n'est située dans leur cône de dominance respectif. En revanche, les alternatives x et y ne sont pas Pareto-optimales car les points z et w sont respectivement dans leur cône de dominance (plus précisément, nous avons $z \succ_P x$ et $w \gg_P y$). Dans cette situation, nous avons donc* $\mathrm{ND}(\{v, w, x, y, z\}) = \{v, w, z\}$. *Par ailleurs, comme nous avons $\neg(z \gg_P x)$ car z n'est pas à l'intérieur du cône de x, alors nous en déduisons que* $\mathrm{SND}(\{v, w, x, y, z\}) = \{v, w, z, x\}$.

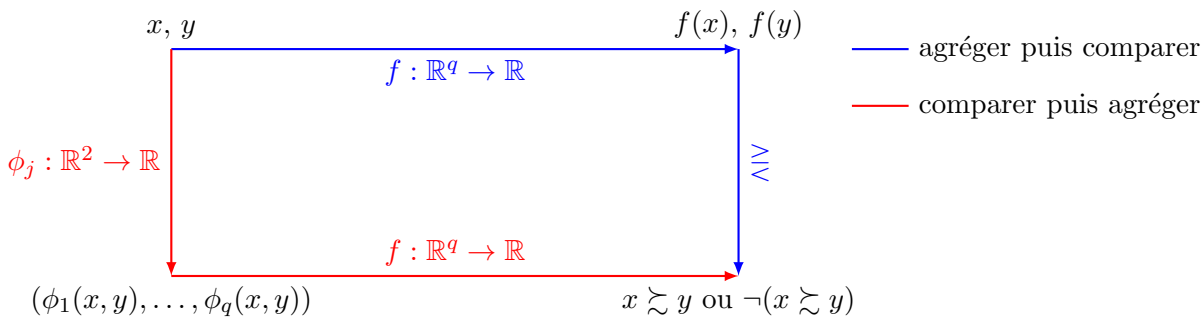
Ce concept d'optimalité a été introduit par le sociologue et économiste italien Vilfredo Pareto pour définir un état de la société dans lequel il n'est pas possible d'améliorer le bien-être d'un individu (ici la performance sur un critère) sans détériorer celui d'un autre. Bien que relativement séduisante, la dominance de Pareto est généralement insuffisante pour pouvoir prendre une décision car celle-ci peut laisser un grand nombre d'alternatives incomparables entre elles. En effet, il suffit que x soit strictement meilleure que y sur un critère et que y soit strictement meilleure que x sur un autre critère pour

que ces deux alternatives ne soient plus comparables. En optimisation combinatoire multicritère, des exemples classiques montrent que toutes les alternatives du problème, aussi nombreuses soient-elles, peuvent être Pareto-optimales (e.g., [Hansen, 1980] pour le problème de plus court chemin multicritère, [Hamacher and Ruhe, 1994] pour le problème d'arbre couvrant multicritère). De ce fait, la détermination de l'ensemble $\mathrm{ND}(\mathcal{X})$ peut s'avérer être une tâche relativement difficile en pratique. Pour contourner ces difficultés, il s'agit très souvent de définir une relation de préférence $\succsim$ plus riche que la dominance de Pareto, permettant de représenter les préférences subjectives du décideur. Une façon classique de procéder est de s'intéresser à la manière dont le décideur définit la résultante des performances sur les différents critères, aussi appelée *phase d'agrégation multicritère*.

L'agrégation est une tâche consistant à combiner différents objets de même type (e.g., évaluations partielles, relations binaires) pour obtenir un nouvel objet représentant la synthèse de ces derniers. Par exemple, en agrégation de préférences, il s'agit généralement de combiner des relations décrivant les préférences de différents agents afin de produire une relation spécifiant la préférence collective. En agrégation multicritère, nous nous intéressons à la combinaison des performances sur les différents critères pour construire une relation de préférence $\succsim$ telle que $x \succsim y$ signifie que le décideur préfère l'alternative x à l'alternative y. Dans la littérature sur l'agrégation multicritère, nous pouvons distinguer les deux approches suivantes pour la construction de la relation $\succsim$ (e.g., [Perny, 2000, Grabisch and Perny, 2003]) :

- *Agréger puis comparer :* cette approche consiste essentiellement à attribuer une évaluation globale à chaque alternative en utilisant une fonction d'agrégation $f : \mathbb{R}^q \to \mathbb{R}$, puis de comparer les évaluations scalaires $f(x)$ et $f(y)$ pour déterminer si x est meilleure que y ou non. Très souvent, la relation de préférence $\succsim$ est définie comme suit : $x \succsim y$ si et seulement si $f(x) \geq f(y)$. Notons que cette approche présuppose la commensurabilité des différents critères pour que la comparaison de ces derniers à travers la fonction d'agrégation ait du sens.
- *Comparer puis agréger :* pour cette approche, il s'agit tout d'abord de comparer x et y critère par critère à l'aide des fonctions $\phi_j : \mathbb{R}^2 \to \mathbb{R}$, puis d'agréger les indices de préférences $\phi_j(x_j, y_j)$ avec une fonction d'agrégation $f : \mathbb{R}^q \to \mathbb{R}$ pour finalement déduire si $x \succsim y$ ou non. Remarquons que la commensurabilité des critères n'est pas requise ici puisque ces derniers ne sont jamais directement comparés les uns aux autres.

La figure suivante permet d'illustrer graphiquement la différence entre ces deux approches :



Dans la littérature, il existe de nombreuses fonctions d'agrégation, généralement justifiées par des propriétés mathématiques qu'il convient de vérifier dans certaines situations (démarche axiomatique). Sans prétendre à l'exhaustivité, nous donnons ci-après quelques propriétés mathématiques plus ou moins désirables selon le contexte décisionnel étudié :

Définition 6 (idempotence)**.** *Une fonction* $f : \mathbb{R}^q \to \mathbb{R}$ *est* idempotente *si et seulement si :*

$$\forall a \in \mathbb{R}, f(a, \ldots, a) = a$$

Cette propriété peut être vue comme le respect de l'unanimité dans des contextes où les différents critères représentent l'utilité de différents agents. Dans le cadre de la décision multicritère, cela signifie que la valeur obtenue par agrégation ne doit pas être différente des composantes du vecteur si celles-ci sont toutes égales.

Définition 7 (monotonie)**.** *Une fonction* $f : \mathbb{R}^q \to \mathbb{R}$ *est* monotone *si et seulement si :*

$$\forall x, y \in \mathbb{R}^q, x \succ_P y \Rightarrow f(x) \geq f(y)$$

La monotonie est dite *stricte* si et seulement si cette dernière inégalité est toujours stricte. Dans le cadre de l'approche *agréger puis comparer*, imposer la monotonie revient à assurer la Pareto-optimalité, qui est une notion d'optimalité compatible avec les préférences subjectives de tout agent rationnel.

Définition 8 (compromis)**.** *Une fonction* $f : \mathbb{R}^q \to \mathbb{R}$ *est de* compromis *si et seulement si :*

$$\forall x \in \mathbb{R}^q, \min_{j \in \mathcal{Q}} x_j \leq f(x) \leq \max_{j \in \mathcal{Q}} x_j$$

Cette propriété mathématique permet de traduire un effet compensatoire entre les différentes composantes du vecteur à agréger.

Définition 9 (stabilité par changement d'échelle linéaire)**.** *Une fonction* $f : \mathbb{R}^q \to \mathbb{R}$ *est* stable par changement d'échelle linéaire *si et seulement si :*

$$\forall x \in \mathbb{R}^q, \forall \lambda > 0, \forall \gamma \in \mathbb{R}, f(\lambda x_1 + \gamma, \ldots, \lambda x_q + \gamma) = \lambda f(x) + \gamma$$

En particulier, nous parlons de fonction *homogène* lorsque $\gamma = 0$. Cette propriété impose que les préférences restent inchangées après un changement d'échelle linéaire, ce qui est une contrainte relativement naturelle. Par exemple, il semble raisonnable d'imposer que le classement relatif des élèves d'une classe ne change pas après avoir identiquement changé l'échelle des notes dans les différentes matières.

Définition 10 (symétrie)**.** *Une fonction* $f : \mathbb{R}^q \to \mathbb{R}$ *est* symétrique *si et seulement si :*

$$\forall x \in \mathbb{R}^q, \forall \theta \in \Theta(\mathcal{Q}), f(x_1, \ldots, x_q) = f(x_{\theta(1)}, \ldots, x_{\theta(q)})$$

où $\Theta(\mathcal{Q})$ *est l'ensemble de toutes les permutations sur* $\mathcal{Q}$*.*

La symétrie n'est généralement pas souhaitée en décision multicritère, le décideur n'accordant pas toujours la même importance aux différents critères. En revanche, lorsque les critères représentent le point de vue de plusieurs individus, cette propriété est indispensable pour des raisons d'équité.

Il existe des relations entre les différentes propriétés mathématiques que nous venons de présenter, comme par exemple les suivantes :

- Toute fonction de compromis est idempotente.
- Toute fonction idempotente et monotone est de compromis.
- Toute fonction stable par changement d'échelle linéaire est idempotente.

D'autres propriétés mathématiques peuvent bien évidemment être exigées selon le contexte décisionnel, comme par exemple la continuité de la fonction d'agrégation en optimisation continue. Dans la sous-section suivante, nous présentons les différentes fonctions d'agrégation que nous avons étudiées durant cette thèse, en précisant à chaque fois si les propriétés que nous avons énoncées sont vérifiées par celles-ci.

1.1.2 Quelques opérateurs d'agrégation

Cette sous-section est consacrée à la présentation des fonctions d'agrégation paramétriques que nous avons étudiées durant cette thèse. Pour simplifier la présentation, nous nous plaçons ici dans le cadre de l'approche *agréger puis comparer*. Cette approche permet de dériver une relation $\succsim$ modélisant les préférences du décideur par comparaison des *utilités* (performances globales) des alternatives ; l'utilité de chaque alternative est obtenue par application de la fonction d'agrégation à son vecteur de performances. Dans le cadre de cette thèse, nous nous sommes plus particulièrement intéressés aux modèles décisionnels fondés sur les rang. Pour définir l'utilité d'une alternative, les opérateurs d'agrégation fondés sur le rang commencent par trier en ordre croissant les composantes de son évaluation vectorielle avant de réaliser l'agrégation. De cette façon, les modèles fondés sur le rang permettent d'associer des poids aux rangs au lieu des critères, ce qui offre notamment la possibilité de contrôler l'importance attribuée à la plus mauvaise performance, à la meilleure, ou à toute autre statistique d'ordre (minimum, maximum, médiane...). Pour présenter ces modèles, nous utiliserons la notation $(.)$ pour toute alternative $x \in \mathcal{X}$ afin de faire référence à une permutation des critères $\mathcal{Q} = \{1, \ldots, q\}$ vérifiant $x_{(1)} \leq \ldots \leq x_{(q)}$.

La somme pondérée

La somme pondérée, notée SP, fait partie des modèles paramétriques les plus simples en agrégation multicritère. Les paramètres de ce modèle sont sous la forme d'un vecteur de poids strictement positifs $\omega = (\omega_1, \ldots, \omega_q)$ tel que $\sum_{j \in \mathcal{Q}} \omega_j = 1$. Chaque poids ω_j permet de quantifier l'importance du critère $j \in \mathcal{Q}$ pour le décideur. Plus précisément, étant donné un vecteur ω, l'utilité d'une alternative $x \in \mathcal{X}$ aux yeux du décideur est définie comme suit :

Définition 11 (somme pondérée)**.** $\mathrm{SP}(x, \omega) = \sum_{j \in Q} \omega_j x_j$

Par définition, la relation de préférence $\succsim$ induite par la somme pondérée est la suivante :

$$x \succsim y \;\Leftrightarrow\; \sum_{j \in Q} \omega_j x_j \geq \sum_{j \in Q} \omega_j y_j$$

Il est facile de vérifier que la somme pondérée est un opérateur monotone, autrement dit que l'inégalité $\mathrm{SP}(x,\omega) > \mathrm{SP}(y,\omega)$ est vraie dès lors que $x \succ_P y$ (cf. définition 7). Pour le montrer, rappelons que $x \succ_P y$ signifie que $x_j \geq y_j$ pour tout $j \in \mathcal{Q}$ et qu'il existe $j \in \mathcal{Q}$ tel que $x_j > y_j$. Comme $\omega_j > 0$ pour tout $j \in \mathcal{Q}$, alors nous en déduisons que $\omega_j x_j \geq \omega_j y_j$ pour tout $j \in \mathcal{Q}$ et qu'il existe $j \in \mathcal{Q}$ tel que $\omega_j x_j > \omega_j y_j$. Le résultat est ensuite obtenu en sommant toutes ces inégalités. Par ailleurs, il est tout aussi simple de montrer que la somme pondérée satisfait la propriété de stabilité par changement d'échelle linéaire, c'est-à-dire que l'égalité $\mathrm{SP}((\lambda x_1 + \gamma, \ldots, \lambda x_q + \gamma), \omega) = \lambda\,\mathrm{SP}(x,\omega) + \gamma$ est vraie pour tout $x \in \mathcal{X}$, tout $\lambda > 0$ et tout $\gamma \in \mathbb{R}$ (cf. définition 9). En effet, le résultat est établi par la séquence d'égalités suivante :

$$\mathrm{SP}((\lambda x_1+\gamma,\ldots,\lambda x_q+\gamma),\omega)=\sum_{j\in Q}\omega_j(\lambda x_j+\gamma)=\sum_{j\in Q}\omega_j\lambda x_j+\sum_{j\in Q}\omega_j\gamma=\lambda\sum_{j\in Q}\omega_j x_j+\gamma\sum_{j\in Q}\omega_j=\lambda\,\mathrm{SP}(x,\omega)+\gamma$$

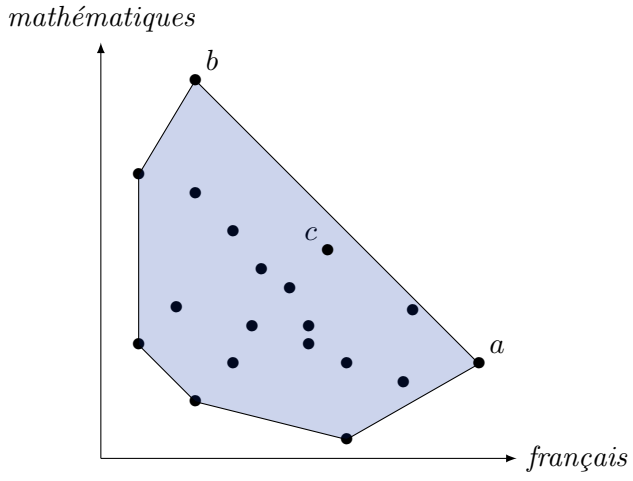
Enfin, la somme pondérée vérifie aussi la propriété de compromis qui est, rappelons-le, définie par la contrainte $\min_{k\in\mathcal{Q}}\{x_k\} \leq \mathrm{SP}(x,\omega) \leq \max_{k\in\mathcal{Q}}\{x_k\}$ pour tout $x \in \mathcal{X}$ (cf. définition 8). En effet :

$$\mathrm{SP}(x,\omega) = \sum_{j\in Q}\omega_j x_j \geq \sum_{j\in Q}\omega_j \min_{k\in\mathcal{Q}}\{x_k\} = \min_{k\in\mathcal{Q}}\{x_k\}\sum_{j\in Q}\omega_j = \min_{k\in\mathcal{Q}}\{x_k\}\times 1 = \min_{k\in\mathcal{Q}}\{x_k\}$$

$$\mathrm{SP}(x,\omega) = \sum_{j\in Q}\omega_j x_j \leq \sum_{j\in Q}\omega_j \max_{k\in\mathcal{Q}}\{x_k\} = \max_{k\in\mathcal{Q}}\{x_k\}\sum_{j\in Q}\omega_j = \max_{k\in\mathcal{Q}}\{x_k\}\times 1 = \max_{k\in\mathcal{Q}}\{x_k\}$$

Du fait de sa simplicité, ce modèle décisionnel est très souvent utilisé en pratique pour agréger des performances. C'est par exemple sur la base de la moyenne des notes que sont évalués les élèves dans le système scolaire français. Néanmoins, cet opérateur se révèle très peu expressif en pratique, ne permettant pas toujours de modéliser des préférences pour les alternatives au profil équilibré :

Exemple 2. *À la fin du premier trimestre scolaire, les élèves d'une classe ont obtenu diverses notes en français (premier critère) et en mathématiques (deuxième critère). Supposons que les étudiants se répartissent dans l'espace des critères de la manière suivante :*



En particulier, les élèves a, b et c ont obtenus les notes suivantes : $a = (20, 5)$, $b = (5, 20)$ et $c = (12, 11)$. Autrement dit, les élèves a et b ont obtenu la note maximale dans l'une des matières, mais tous deux semblent avoir de véritables lacunes dans l'autre matière, ce qui pourrait compromettre leur réussite scolaire. Quant à l'élève c, celui-ci a obtenu des notes relativement correctes dans les deux matières et semble donc avoir des bases solides pour pouvoir continuer cette année scolaire sereinement. Par ailleurs, ni $a = (20, 5)$ ni $b = (5, 20)$ ne Pareto-dominent $c = (12, 11)$. Néanmoins, dès lors que l'on considère une somme pondérée pour les départager, l'élève c ne peut pas être le premier du classement (qu'importe le vecteur de poids $\omega = (\omega_1, \omega_2)$ utilisé). Ce résultat s'explique par le fait que le point $c = (12, 11)$ est situé à l'intérieur de l'enveloppe convexe de tous les points (zone bleue), et que seuls des points sur les bords de l'enveloppe peuvent être obtenus en maximisant une somme pondérée.

Cet exemple illustre le fait que choisir la somme pondérée comme fonction d'agrégation peut condamner des solutions Pareto-optimales a priori, c'est-à-dire avant même d'avoir décidé des poids à utiliser pour pondérer les différents critères. Pourtant, le décideur peut raisonnablement juger que ces alternatives réalisent des compromis intéressants entre les différents critères. Ces limites bien connues de la somme pondérée justifient le fait que les chercheurs en aide à la décision s'intéressent à d'autres agrégateurs, comme ceux fondés sur le rang.

La moyenne pondérée ordonnée

La moyenne pondérée ordonnée (notée OWA pour *Ordered Weighted Averaging* en anglais) (e.g., [Yager, 1988]) fait partie des agrégateurs les plus simples parmi ceux fondés sur le rang. Plus précisément, pour définir l'utilité d'une alternative au sens de OWA, il s'agit d'appliquer une somme pondérée sur son vecteur de performances après avoir trié ses composantes en ordre croissant. Ainsi, les paramètres de ce modèle décisionnel sont sous la forme d'un vecteur de poids positifs $\omega = (\omega_1, \ldots, \omega_q)$ de somme égale à 1, où le poids ω_j représente l'importance du $j^{\text{ème}}$ rang aux yeux du décideur. Plus formellement, l'utilité d'une alternative $x \in \mathcal{X}$ au sens de OWA est donnée par :

Définition 12 (Ordered Weighted Averaging)**.** $\text{OWA}(x, \omega) = \sum_{j \in \mathcal{Q}} \omega_j x_{(j)}$

En d'autres termes, $\text{OWA}(x, \omega)$ est le produit scalaire entre le vecteur de poids ω et le vecteur trié $(x_{(1)}, \ldots, x_{(q)})$. La relation de préférence $\succsim$ induite par ce modèle décisionnel est définie comme suit :

$$x \succsim y \;\Leftrightarrow\; \sum_{j \in \mathcal{Q}} \omega_j x_{(j)} \geq \sum_{j \in \mathcal{Q}} \omega_j y_{(j)}$$

Cette famille de fonctions d'agrégation inclut en particulier la moyenne arithmétique, obtenue en prenant tout simplement $\omega_j = 1/q$ pour tout $j \in \mathcal{Q}$. Dans cette famille, nous retrouvons aussi toutes les statistiques d'ordre en considérant un vecteur ω qui est nul sur toutes ses composantes sauf une. Par exemple, les fonctions de minimum et de maximum sont retrouvées avec $\omega_1 = 1$ et $\omega_q = 1$ respectivement.

Par définition, les agrégateurs de type OWA sont symétriques (cf. définition 10), car l'agrégation est réalisée sur les vecteurs de performances triés. Rappelons que cette propriété est particulièrement désirée

dans les problèmes de décision multicritère où les critères représentent le point de vue de différents agents. Les agrégateurs de type OWA vérifient aussi la propriété de monotonie (cf. définition 7) car ces derniers sont monotones croissants par rapport à chaque composante (e.g., [Yager, 1988]). Par ailleurs, comme l'ordre des composantes ne change pas par application d'une fonction strictement croissante et que le produit scalaire est un opérateur linéaire, alors il est facile de voir que les agrégateurs de type OWA sont aussi stables par changement d'échelle linéaire (cf. définition 9). Enfin, avec des arguments similaires à ceux utilisés pour la somme pondérée, il est possible de montrer que les agrégateurs de type OWA sont des opérateurs de compromis (e.g., [Yager, 1988]).

En théorie du choix social, le modèle OWA est souvent utilisé avec des poids ω_j décroissants comme mesure des inégalités, notamment du fait de la propriété suivante :

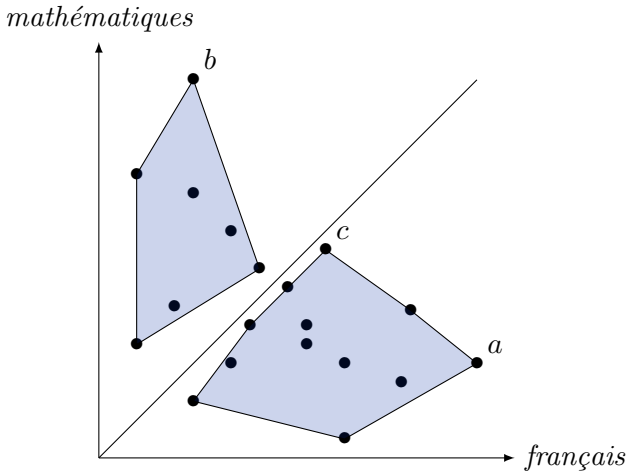
Proposition 1. *Pour tout $x \in \mathbb{R}^q$ et tous $k, l \in \mathcal{Q}$ tels que $k < l$ et $x_l < x_k$, nous avons :*

$$\Big[\forall j \in \{1, \ldots, q-1\}, \omega_j > \omega_{j+1}\Big] \Rightarrow \forall \varepsilon \in \,]0, x_k - x_l[, \mathrm{OWA}(x, \omega) < \mathrm{OWA}(x^\varepsilon, \omega)$$

où $x^\varepsilon = (x_1, \ldots, x_k - \varepsilon, \ldots, x_l + \varepsilon, \ldots, x_q)$.

En d'autres termes, si les poids $\omega_j, j \in \mathcal{Q}$, sont strictement décroissants, alors passer d'une alternative à une autre en effectuant un "équilibrage" d'amplitude ε, permettant de réduire les inégalités entre les individus k et l, augmente nécessairement l'utilité au sens de OWA. Ces équilibrages sont connus sous le nom de transferts de Pigou-Dalton en économie et sont souvent utilisés pour exprimer l'idée d'équité (cf. e.g., [Shorrocks, 1983, Weymark, 1981]). Puisque les agrégateurs de type OWA sont monotones, alors les alternatives optimales au sens de OWA sont les solutions Pareto-optimales qui ne peuvent être améliorées par des transferts de Pigou-Dalton (dans le cas de poids décroissants). Une autre façon de se rendre compte que les agrégateurs de type OWA permettent de favoriser les alternatives au profil équilibré lorsque les poids sont décroissants, c'est de remarquer que nous accordons plus d'importance à la plus mauvaise performance, un peu moins à la seconde plus mauvaise, et ainsi de suite jusqu'à la meilleure. L'exemple suivant permet d'illustrer cette propriété d'équité :

Exemple 3. *Considérons le vecteur de poids $\omega = (0.6, 0.4)$ pour évaluer les étudiants $a = (20, 5)$, $b = (5, 20)$ et $c = (12, 11)$ de l'exemple 2. On a alors* $\mathrm{OWA}(a, \omega) = \mathrm{OWA}(b, \omega) = 0.6 \times 5 + 0.4 \times 20 = 11 < \mathrm{OWA}(c, \omega) = 0.6 \times 11 + 0.4 \times 12 = 11.4$. *Ainsi, l'étudiant c, qui a un profil plus équilibré que les deux autres étudiants, est ici considéré comme le meilleur des trois. Rappelons que ce phénomène ne peut pas se produire avec une somme pondérée, puisque le point c est situé à l'intérieur de l'enveloppe convexe de tous les points. Sur le graphique suivant, nous constatons que le point c est en réalité situé sur les bords de l'enveloppe convexe des points (x_1, x_2) vérifiant $x_2 \leq x_1$. Autrement dit, le point c est sur les bords de l'enveloppe convexe des points subissant la même permutation des critères durant l'agrégation. Il s'avère que seuls de tels points peuvent maximiser une fonction de type* OWA*, car celle-ci n'est rien d'autre qu'une somme pondérée sur les différents sous-espaces (définis par les permutations possibles).*



Finalement, nous avons vu que le modèle OWA permet de modéliser des décideurs ayant une préférence pour les alternatives équilibrées (en utilisant des poids décroissants) tout en assurant la Pareto-optimalité. Ce modèle décisionnel est très souvent utilisé en décision multi-objectifs dans des situations où les différents objectifs représentent les points de vue de plusieurs agents, comme par exemple dans les problèmes d'allocation équitable (e.g., [Korhonen et al., 1990, Ogryczak, 2000]) ou dans les procédures de vote (e.g. [Goldsmith et al., 2014, Elkind and Ismaili, 2015]).

Néanmoins, en décision multicritère (mono-agent), les critères n'ont généralement pas la même importance pour le décideur, et donc il n'est pas souhaitable de faire usage d'un modèle décisionnel symétrique comme l'est OWA. Dans la littérature, on retrouve des extensions du modèle OWA tenant compte à la fois des performances des alternatives et de l'importance des critères. Parmi les modèles décisionnels proposés, citons par exemple la moyenne ordonnée doublement pondérée [Torra, 1997] ou encore la moyenne pondérée hybride [Llamazares, 2011], qui sont des modèles paramétrés par deux vecteurs : un vecteur de poids associés aux rangs et un vecteur de poids rattachés aux critères. Dans le cadre de cette thèse, nous nous intéresserons plus particulièrement aux intégrales de Choquet car celles-ci étendent les possibilités descriptives offertes par le modèle OWA en associant un poids à chaque sous-ensemble de critères.

L'intégrale de Choquet

Les intégrales de Choquet (e.g., [Choquet, 1953, Schmeidler, 1986, Grabisch and Labreuche, 2010]) forment une famille de fonctions d'agrégation des plus appréciées dans le domaine de l'aide à la décision car celle-ci offre la possibilité de modéliser des synergies positives comme négatives entre critères, avec différents niveaux d'intensité. En effet, les paramètres de ce modèle sont sous la forme d'une capacité, c'est-à-dire une fonction définie sur l'ensemble des parties de $\mathcal{Q}$, monotone croissante avec l'inclusion, associant un poids à chaque sous-ensemble de critères. Plus formellement :

Définition 13 (capacité)**.** *La fonction* $v : 2^{\mathcal{Q}} \to [0, 1]$ *est une capacité normalisée si et seulement si :*

- $v(\emptyset) = 0$, $v(\mathcal{Q}) = 1$ *(normalisation),*
- $v(A) \leq v(B)$ *pour tous* $A \subset B \subseteq \mathcal{Q}$ *(monotonie).*

Signalons que la contrainte de monotonie admet la formulation alternative suivante : $v(A) \leq v(A \cup \{j\})$ pour tout $A \subset \mathcal{Q}$ et tout $j \in \mathcal{Q} \backslash A$.

Définition 14. *Une capacité $v : 2^{\mathcal{Q}} \to [0,1]$ est dite :*

- *convexe (ou super-modulaire) si $v(A \cup B) + v(A \cap B) \geq v(A) + v(B)$ pour tous $A, B \subseteq \mathcal{Q}$,*
- *additive si $v(A \cup B) + v(A \cap B) = v(A) + v(B)$ pour tous $A, B \subseteq \mathcal{Q}$,*
- *concave (ou sous-modulaire) si $v(A \cup B) + v(A \cap B) \leq v(A) + v(B)$ pour tous $A, B \subseteq \mathcal{Q}$.*

Nous verrons que ces propriétés vont permettre à l'intégrale de Choquet de modéliser des comportements différents vis-à-vis de l'équité entre les critères. Avant cela, pour pouvoir donner une formulation précise de l'intégrale de Choquet, nous introduisons les ensembles de critères suivants :

Définition 15 (ensemble de niveau)**.** *Le $j^{ème}$ ensemble de niveau d'une alternative x donnée, noté $X_{(j)}$, est défini par : $X_{(j)} = \{(j), \ldots, (q)\}$.*

À partir de ces ensembles de critères, nous pouvons définir l'utilité d'une alternative $x \in \mathcal{X}$ au sens de l'intégrale de Choquet comme suit :

Définition 16 (intégrale de Choquet)**.**

$$\mathrm{Ch}(x, v) = \sum_{j \in \mathcal{Q}} \left[x_{(j)} - x_{(j-1)}\right] v(X_{(j)}), \ \textit{avec } x_{(0)} = 0 \tag{1.1}$$

$$= \sum_{j \in \mathcal{Q}} \left[v(X_{(j)}) - v(X_{(j+1)})\right] x_{(j)}, \ \textit{avec } X_{(q+1)} = \emptyset \tag{1.2}$$

En utilisant l'équation (1.1), nous pouvons définir la relation de préférence $\succsim$ induite par l'intégrale de Choquet de la manière suivante :

$$x \succsim y \ \Leftrightarrow \ \sum_{j \in \mathcal{Q}} \left[x_{(j)} - x_{(j-1)}\right] v(X_{(j)}) \geq \sum_{j \in \mathcal{Q}} \left[y_{(j)} - y_{(j-1)}\right] v(Y_{(j)})$$

L'intégrale de Choquet est un opérateur idempotent (cf. définition 6), stable par changement d'échelle linéaire (cf. définition 9), monotone (cf. définition 7) et de compromis (cf. définition 8) [Marichal, 2000]. Pour plus de détails sur la justification axiomatique de ce modèle décisionnel, le lecteur est invité à se référer aux travaux de Schmeidler [1986], Marichal [2000], Köbberling and Wakker [2003], Labreuche [2012] et de Timonin [2016].

Intéressons-nous maintenant à la modélisation de préférences en faveur de l'équité avec une intégrale de Choquet. Lorsque v est convexe (cf. définition 14), nous avons $\mathrm{Ch}(\lambda x + (1-\lambda)y, v) \geq \lambda\, \mathrm{Ch}(x, v) + (1-\lambda)\, \mathrm{Ch}(y, v)$ pour toutes alternatives $x, y \in \mathcal{X}$ et tout réel $\lambda \in [0,1]$, ce qui signifie que l'intégrale de Choquet est une fonction concave (e.g., [Lovász, 1983]). De ce fait, l'usage d'une capacité convexe permet de favoriser l'équité, comme le suggère la proposition suivante (e.g., [Chateauneuf et al., 1999]) :

Proposition 2. $\forall x^1, x^2, \dots x^n \in \mathbb{R}^q$, $\forall \lambda = (\lambda_1, \dots, \lambda_n) \in [0,1]^n$, $\sum_{i=1}^n \lambda_i = 1$ *:*

$$\mathrm{Ch}(x^1, v) = \ldots = \mathrm{Ch}(x^n, v) \;\Rightarrow\; \forall k \in \{1, 2, \dots, n\},\; \mathrm{Ch}(\sum_{i=1}^{n} \lambda_i x^i, v) \geq \mathrm{Ch}(x^k, v)$$

En d'autres termes, avec une capacité convexe, si le décideur est indifférent entre plusieurs alternatives $x^1, \dots, x^n$, alors nous savons qu'il préfère, à chacune d'elle, toute combinaison convexe de ces alternatives (représentant une sorte de compromis). À titre d'exemple, si le décideur est indifférent entre les alternatives $x^1 = (0,1)$ et $x^2 = (1,0)$, alors celui-ci préfère l'alternative $0.5x^1 + 0.5x^2 = (0.5, 0.5)$ à ces deux dernières puisque nous avons $\mathrm{Ch}(0.5x^1 + 0.5x^2, v) \geq \mathrm{Ch}(x^1, v)$ et $\mathrm{Ch}(0.5x^1 + 0.5x^2, v) \geq \mathrm{Ch}(x^2, v)$ (d'après la proposition 2). Sans grande surprise, nous obtenons la préférence contraire lorsque la capacité est concave (cf. définition 14). En effet, nous avons alors $\mathrm{Ch}(\lambda x + (1-\lambda)y, v) \leq \lambda\,\mathrm{Ch}(x, v) + (1-\lambda)\,\mathrm{Ch}(y, v)$ pour toutes alternatives $x, y \in \mathcal{X}$ et tout $\lambda \in [0,1]$, ce qui signifie que l'intégrale de Choquet est une fonction convexe (e.g., [Lovász, 1983]). Plus généralement, considérer des capacités ni convexes ni concaves permet de modéliser des préférences plus complexes. La famille des intégrales de Choquet est en fait une généralisation de nombreux modèles décisionnels, et nous retrouvons en particulier :

- les sommes pondérées, en considérant une capacité v additive (cf. définition 14).
- les opérateurs de type OWA [Yager, 1988], en prenant v symétrique, c'est-à-dire définie par $v(A) = \phi(|A|)$ pour tout $A \subseteq \mathcal{Q}$, où ϕ est une fonction croissante vérifiant $\phi(0) = 0$ et $\phi(1) = 1$.
- les moyennes ordonnées doublement pondérées [Torra, 1997], en prenant une capacité v définie par $v(A) = \phi(\sum_{j \in A} p_j)$ pour tout $A \subseteq \mathcal{Q}$, où $(p_1, \dots, p_q)$ est un vecteur de pondération des critères et ϕ est croissante telle que $\phi(0) = 0$ et $\phi(1) = 1$.

Dans certaines situations, il peut être utile de considérer une autre formulation bien connue de l'intégrale de Choquet, fondée sur l'inverse de Möbius associée à la capacité (e.g., [Rota, 1964]) :

Définition 17 (inverse de Möbius)**.** *L'inverse de Möbius* $m : 2^{\mathcal{Q}} \to \mathbb{R}$ *associée à une fonction d'ensemble* $v : 2^{\mathcal{Q}} \to \mathbb{R}$ *est définie par :*

$$\forall A \subseteq \mathcal{Q},\; m(A) = \sum_{A' \subseteq A} (-1)^{|A \setminus A'|} v(A')$$

Les valeurs $m(A)$, $A \subseteq \mathcal{Q}$, sont communément appelées *masses de Möbius*. Toute capacité v admet alors la formulation alternative suivante en terme d'inverse de Möbius :

$$\forall A \subseteq \mathcal{Q},\; v(A) = \sum_{A' \subseteq A} m(A') \tag{1.3}$$

Avec cette formulation, il est facile de voir que toute capacité v dont les masses de Möbius sont non-négatives (aussi appelée *fonction de croyance*) est nécessairement convexe (e.g., [Shafer, 1976]). Finalement, l'utilité d'une alternative $x \in \mathcal{X}$ au sens de l'intégrale de Choquet peut s'écrire en fonction des masses de Möbius de la façon suivante (e.g., [Chateauneuf and Jaffray, 1989]) :

Définition 18 (intégrale de Choquet)**.** $\mathrm{Ch}(x, v) = \sum_{A \subseteq \mathcal{Q}} m(A) \min_{j \in A} x_j$

Cette définition donne une autre interprétation de l'intégrale de Choquet : c'est une somme pondérée dans un espace à 2^q dimensions dont le jeu de poids est l'ensemble des masses de Möbius. Cette somme devient plus compacte en considérant ce que l'on appelle des capacités *k-additives* (e.g., [Grabisch, 1996, Grabisch et al., 2009]) :

Définition 19 (capacité k-additive)**.** *Une capacité v est dite k-additive si son inverse de Möbius m satisfait les conditions suivantes :*

- $m(A) = 0$ *pour tout ensemble* $A \subseteq \mathcal{Q}$ *tel que* $|A| > k$,
- $m(A) \neq 0$ *pour au moins un ensemble* $A \subseteq \mathcal{Q}$ *tel que* $|A| = k$.

Pour $k = 1$, nous retombons trivialement sur les capacités additives (cf. définition 14). Pour des petites valeurs de k, les capacités k-additives sont très intéressantes car, avec un nombre réduit de paramètres, celles-ci permettent de modéliser des synergies entre les critères (plus précisément, entre au plus k critères). À titre d'exemple, une capacité 2-additive est complètement caractérisée avec seulement $(q^2 + q)/2$ valeurs (une masse de Möbius par singleton et par paire de critères) au lieu des 2^q valeurs nécessaires dans le cas d'une capacité quelconque. De plus, avec des capacités 2-additives, il est possible de modéliser les interactions suivantes :

- $\forall j, k \in \mathcal{Q}, m(\{j,k\}) > 0 \Rightarrow v(\{j,k\}) > v(\{j\}) + v(\{k\})$ (interaction positive),
- $\forall j, k \in \mathcal{Q}, m(\{j,k\}) = 0 \Rightarrow v(\{j,k\}) = v(\{j\}) + v(\{k\})$ (aucune interaction),
- $\forall j, k \in \mathcal{Q}, m(\{j,k\}) < 0 \Rightarrow v(\{j,k\}) < v(\{j\}) + v(\{k\})$ (interaction négative).

Dans cette section, nous avons étudié la question de la modélisation des préférences pour des problèmes de décision multicritère. Dans la section suivante, nous nous intéressons à la problématique de modélisation de préférences collectives pour des problèmes de prise de décision multi-agents.

1.2 Décision collective

La prise de décision collective peut intervenir dans des situations où des individus se rassemblent en un groupe pour faire des choix ensemble. À titre d'exemple, on peut penser à un comité de sélection devant choisir une personne pour un poste particulier ou encore à une entreprise devant déterminer un plan d'action marketing et commercial. La nécessité de prendre des décisions collectivement peut bien évidemment surgir dans des problèmes à plus grande échelle, comme par exemple durant les élections présidentielles et législatives françaises. Les problèmes de décision collective sont assez proches des problèmes de décision multicritère car il s'agit de prendre en compte plusieurs points de vue simultanément. En particulier, ces deux problématiques se rejoignent lorsque chaque alternative peut être associée à un vecteur dont les composantes représentent son utilité aux yeux des différents agents. De ce fait, il est facile de se rendre compte que la prise de décision collective soulève des problèmes d'agrégation assez similaires à ceux rencontrés en décision multicritère. Soulignons toutefois que nos exigences sur la méthode d'agrégation ne sont pas exactement les mêmes dans ces deux contextes décisionnels. Par exemple, la méthode d'agrégation doit très souvent être symétrique en décision collective pour des raisons d'équité, alors que cette propriété est indésirable en décision multicritère dès lors que les critères n'ont pas la même importance.

Dans cette section, nous nous écartons de la décision multicritère en considérant uniquement des situations de vote où les préférences des n agents/votants sur l'ensemble $\mathcal{X}$ des alternatives possibles sont observables uniquement sous la forme de relations binaires $\succsim_i$, $i \in N = \{1, \ldots, n\}$. Plus précisément, pour toutes alternatives $x, y \in \mathcal{X}$, nous avons $x \succsim_i y$ si et seulement si l'agent $i \in N$ considère que x est au moins aussi bonne que y. En théorie du vote, il est généralement supposé que les votants sont toujours capables de distinguer deux alternatives et donc que leurs préférences sont en réalité représentables par des relations strictes $\succ_i$, $i \in N$; dans la littérature, la donnée $(\succ_1, \ldots, \succ_n)$ est souvent appelée *profil de préférences* ou *profil* pour une formulation plus courte. Dans ce contexte, la problématique d'agrégation des préférences se distingue un peu plus du problème d'agrégation étudié en décision multicritère. En effet, pour déterminer la meilleure alternative, il ne s'agit plus de résumer les performances de chaque alternative en une unique valeur. Il s'agit à la place de combiner les relations de préférence $\succ_i$, $i \in N$, pour pouvoir prendre une décision de groupe. La prise de décision collective se fait généralement par le biais de ce que l'on appelle une *fonction de choix social* :

Définition 20 (fonction de choix social)**.** *Une fonction de choix social est une fonction qui associe, à tout profil admissible* $(\succ_1, \ldots, \succ_n)$*, un sous ensemble non vide de l'ensemble* $\mathcal{X}$ *des alternatives possibles. Elle est dite* mono-évaluée *si elle retourne toujours une seule alternative.*

Les alternatives retournées par la fonction de choix social représentent les meilleures solutions du point de vue collectif et sont communément appelées les *vainqueurs/gagnants* de l'élection. Dans certaines situations de vote, il peut être intéressant de construire entièrement la relation de préférence sociale au lieu de se contenter de la connaissance des meilleures alternatives. À titre d'exemple, lorsque plusieurs postes sont à pourvoir, il est utile de connaître les premiers candidats du classement correspondant aux préférences des membres du comité de sélection. Une relation de préférence sociale peut être construite, par exemple, en utilisant les scores que la fonction de choix social attribue aux différentes alternatives lors de la combinaison des relations $\succ_i, i \in N$. Plus généralement, une relation de préférence sociale est construite à l'aide de ce que l'on appelle une *fonction d'agrégation des préférences* :

Définition 21 (fonction d'agrégation des préférences)**.** *Une fonction d'agrégation des préférences est une fonction qui associe, à tout profil de préférences admissible* $(\succ_1, \ldots, \succ_n)$*, une relation* $\succsim$ *représentant la préférence sociale.*

Dans la littérature, il existe de nombreuses fonctions de choix social, souvent justifiées par des axiomes exprimant des propriétés que la procédure de vote doit satisfaire. La diversité des procédures de vote existantes s'explique notamment par la difficulté d'en définir une ne présentant aucune mauvaise propriété. Cette difficulté, qui est au cœur des problématiques étudiées en théorie du choix social, va constituer le fil conducteur de cette section. Plus précisément, nous allons commencer par présenter la règle majoritaire, qui est probablement l'une des premières règles de vote qui vient à l'esprit quand on souhaite mettre en place une élection. Après avoir discuté de ses points forts et de ses faiblesses, nous nous intéresserons ensuite à la problématique d'agrégation de préférences de manière plus générale. Enfin, nous terminerons

avec une présentation des règles de vote que nous avons étudiées durant cette thèse, en les situant par rapport à des théorèmes d'impossibilité très connus dans le domaine du choix social.

1.2.1 Une introduction à la théorie du vote : la règle majoritaire

Pour sélectionner une fonction de choix social dans une situation de vote donné, nous pouvons adopter une approche axiomatique en nous restreignant aux fonctions satisfaisant des propriétés jugées adaptées à la situation en question. Par exemple, il peut paraître raisonnable d'exiger que la fonction de choix social vérifie les propriétés suivantes :

- *Universalité* : le domaine de la fonction est l'ensemble de tous les profils possibles, ce qui permet aux votants d'être libres d'opinion et d'expression.
- *Anonymat* : les votants jouent des rôles équivalents, autrement dit l'issue du vote ne doit pas dépendre de l'identité des votants. Cette propriété permet d'assurer qu'aucun votant n'est favorisé par la procédure de vote.
- *Neutralité* : les différentes alternatives sont traitées de manière équivalente durant la procédure de vote, qu'importe leur "nom". Ceci permet de garantir qu'aucune alternative n'est privilégiée par la procédure de vote.
- *Monotonie* : si un agent change son vote en faveur d'une alternative, et que les autres votes demeurent inchangés, alors cette dernière ne doit pas régresser strictement dans le classement final. En particulier, si cette alternative était vainqueur ex æquo, alors celle-ci devient l'unique gagnant de l'élection.

Imposer ces propriétés conduit au résultat suivant :

Théorème 1 ([May, 1952])**.** *Dans le cas où l'ensemble des alternatives possibles ne contient que deux éléments, la règle majoritaire est la seule procédure de vote qui vérifie les axiomes* universalité*,* anonymat*,* neutralité *et* monotonie*.*

Au vu de ce résultat, nous pouvons considérer que la règle majoritaire est la seule règle de vote "raisonnable" dans le cas où seules deux alternatives sont possibles. Ce résultat permet aussi de caractériser les autres règles de vote en fonction des axiomes qu'elles ne vérifient pas. Par exemple, une procédure de vote dictatoriale ne satisfait évidemment pas la propriété *anonymat*. Quant à la propriété *neutralité*, elle n'est pas vérifiée par certaines procédures de votes à majorité qualifiée, notamment lorsqu'une alternative doit recevoir strictement plus de la moitié des voix pour gagner (par exemple 60% des voix), alors que l'autre est considérée comme le choix par défaut. Enfin, la règle de la majorité inversée, choisissant l'alternative rejetée à la majorité, est un exemple simple (bien que farfelu) de procédure ne vérifiant pas la propriété *monotonie*.

Cependant, nous allons voir que l'utilisation de la règle majoritaire avec plus de trois alternatives possibles est beaucoup plus discutable. En effet, intéressons-nous au principe d'optimalité suivant :

Définition 22 (vainqueur de Condorcet)**.** *Un vainqueur de Condorcet est une alternative qui bat à la majorité chaque autre alternative prise individuellement. Si une telle alternative existe, elle est unique.*

Cette notion d'optimalité doit son nom au marquis de Condorcet, mathématicien et philosophe français du XVIII$^{\text{ème}}$ siècle [Condorcet, 1785]. Bien que ce principe d'optimalité soit relativement séduisant, il arrive qu'aucune alternative ne le vérifie, nous laissant dans une situation d'indétermination :

Exemple 4. *Considérons une situation de vote où trois agents ont les préférences suivantes concernant les alternatives x, y et z :*

$$\begin{array}{l} x \succ_1 y \succ_1 z \\ y \succ_2 z \succ_2 x \\ z \succ_3 x \succ_3 y \end{array}$$

Dans cette situation, l'alternative x est socialement préférée à l'alternative y au sens de la règle majoritaire, car 2 votants (les agents 1 et 3) sur 3 préfèrent x à y (ce qui constitue une majorité). De même, nous pouvons déduire que y est socialement préférée à l'alternative z et que z est socialement préférée à x, au sens de la règle majoritaire. Ainsi, nous observons un cycle dans la relation de préférence sociale $\succsim$ induite par la règle majoritaire : nous avons $x \succsim y \succsim z \succsim x$. Il n'existe donc pas de vainqueur de Condorcet dans cette situation.

Cet exemple illustre le célèbre *paradoxe Condorcet* : la relation de préférence sociale induite par la majorité n'est pas nécessairement transitive, et en particulier l'existence d'un vainqueur de Condorcet n'est pas garantie. Ce résultat relativement "négatif" concernant la règle majoritaire a suscité de nombreux travaux dans le domaine du choix social, notamment sur la probabilité d'existence d'un vainqueur de Condorcet (e.g. [Gehrlein, 1983, Regenwetter, 2006]), sur la probabilité qu'une règle de vote donnée élise le vainqueur de Condorcet lorsque celui-ci existe (e.g. [Cervone et al., 2005, Gehrlein et al., 2011]) ou encore sur des restrictions de domaine assurant l'existence d'un vainqueur de Condorcet (e.g. [Abello and Johnson, 1984, Saari, 2009]). Une des restrictions les plus connues est la condition de *single-peakedness*, qui peut se formuler de la manière suivante :

Définition 23 (single-peakedness)**.** *Étant donné un ensemble d'alternatives $\mathcal{X} = \{x^1, \ldots, x^m\}$, un profil $(\succ_1, \ldots, \succ_n)$ est dit* single-peaked *s'il existe une permutation σ de $\{1, \ldots, m\}$ telle que, pour tout agent $i \in N$, il existe $j \in \{1, \ldots, m\}$ vérifiant :*

- $x^{\sigma(k)} \succ_i x^{\sigma(k-1)}$ *pour tout* $1 \leq k \leq j$
- $x^{\sigma(k-1)} \succ_i x^{\sigma(k)}$ *pour tout* $j < k \leq m$

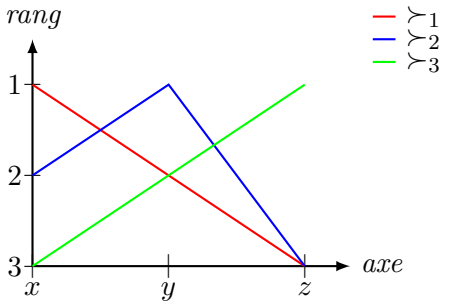
L'axe correspondant à la permutation σ est ici noté $x^{\sigma(1)} > \ldots > x^{\sigma(m)}$.

De manière moins formelle, un profil de préférences est dit *single-peaked* si les alternatives peuvent être alignées sur un axe de sorte que, pour chaque agent, les alternatives deviennent de plus en plus intéressantes en avançant sur l'axe jusqu'à arriver à son alternative préférée, puis deviennent de plus en plus mauvaises. À titre illustratif, nous pouvons penser à une élection politique où les candidats sont ordonnés sur un axe "gauche-droite", ou encore à un problème de décision collective où il s'agit de déterminer une position stratégique sur une route. L'exemple ci-dessous donne une illustration graphique de cette propriété.

Exemple 5. *Considérons un problème de vote impliquant trois agents $N = \{1, \ldots, 3\}$ et trois alternatives notées x, y et z. Avec les préférences de l'exemple 4, il n'existe aucun axe rendant le profil* single-peaked. *En revanche, le profil suivant est* single-peaked *par rapport à l'axe $x > y > z$:*

$$\begin{array}{ccccc} x & \succ_1 & y & \succ_1 & z \\ y & \succ_2 & x & \succ_2 & z \\ z & \succ_3 & y & \succ_3 & x \end{array}$$

Pour nous en convaincre, nous pouvons nous intéresser au graphique suivant :



Sur l'axe des ordonnées, le nombre représente le rang de l'alternative dans l'ordre induit par la relation de préférence $\succ_i$, la valeur 1 correspondant à la meilleure alternative. Pour que le profil soit single-peaked *par rapport à l'axe $x > y > z$, il faut que toutes les courbes soient croissantes puis décroissantes sur ce graphique (ce qui est bien le cas ici).*

L'hypothèse de *single-peakedness* permet de dériver le résultat positif suivant :

Théorème 2 ([Black et al., 1958])**.** *Si le profil de préférences est* single-peaked*, alors la préférence sociale induite par la règle majoritaire est transitive. En particulier, lorsque le nombre d'agents est impair, le vainqueur de Condorcet correspond à la médiane des alternatives préférées des votants.*

Ainsi, l'hypothèse de *single-peakedness* permet de rétablir la transitivité de la règle majoritaire. Par ailleurs, il est facile de tester si un profil est *single-peaked* et de construire un axe associé le cas échéant (e.g. [Escoffier et al., 2008]). Bien que le côté "pratique" de cette restriction de domaine soit très appréciable, il s'avère que les profils observés dans la vie de tous les jours sont rarement *single-peaked*; par exemple, il suffit que trois agents classent en dernière position trois alternatives différentes pour que le profil ne puisse plus être *single-peaked*. Il a toutefois été suggéré que lorsque les agents concentrent leur évaluation sur une dimension commune (cognitive ou idéologique), ces derniers pouvaient induire un profil *single-peaked* (e.g., [Miller, 1992, Dryzek and List, 2004]). Cette hypothèse a récemment été soutenue par des résultats empiriques (e.g., List et al. [2013]), mais mériterait une analyse expérimentale plus poussée.

Une autre limite de la règle majoritaire est que le vainqueur de Condorcet, s'il existe, ne constitue pas nécessairement un bon compromis entre les différentes opinions, comme dans la situation suivante :

Exemple 6. *Considérons un problème de décision collective avec les alternatives* $\mathcal{X} = \{x^1, x^2, \ldots, x^m\}$, $m \geq 3$, *et impliquant* 1000 *agents avec les préférences suivantes :*

$$\forall i \in \{1, \ldots, 499\},\ x^1 \succ_i x^2 \succ_i \ldots \succ_i x^m$$
$$\forall i \in \{500, \ldots, 1000\},\ x^m \succ_i x^2 \succ_i \ldots \succ_i x^1$$

L'alternative x^m *bat à la majorité les* $m-1$ *autres alternatives (avec* 501 *voix sur* 1000*) ; c'est donc le vainqueur de Condorcet. Cependant, certains agents peuvent se sentir complètement lésés suite à l'élection de* x^m *car cette alternative constitue leur dernier choix (*499 *agents sont dans cette situation, soit un peu moins de* 50 *% des votants). Dans cette situation de vote, l'alternative* x^2 *semble être un choix moins contestable que le vainqueur de Condorcet. En effet, l'alternative* x^2 *est classée deuxième par tous, réalisant ainsi un bon compromis entre les différentes opinions.*

Cette exemple illustre le fait que, mis à part le problème de possible indétermination, l'utilisation de la règle majoritaire peut être sujet à discussion lorsque plus de trois alternatives sont à comparer. Cette question a notamment été soulevée par une pensée du célèbre Chat de Philippe Geluck : *"Les règles de la démocratie veulent que ce soit la majorité qui ait raison. Et si la majorité avait tort ? Me direz-vous, ça ne change rien, la majorité aurait raison d'avoir tort."*

1.2.2 Agrégation des préférences : des résultats d'impossibilité

Dans la sous-section précédente, nous avons vu que la règle majoritaire est incontournable lorsque seulement deux alternatives sont à comparer, mais que cette règle n'est pas complètement satisfaisante avec un plus grand nombre d'alternatives. Un des résultats les plus connus dans le domaine de la théorie du choix social est le théorème d'impossibilité d'Arrow, du nom de l'économiste Kenneth Arrow, qui peut être vu comme la preuve de non-existence de processus de choix social "indiscutable". Afin d'énoncer plus précisément ce résultat, considérons les axiomes suivants :

- *Unanimité (ou Pareto optimalité)* : si tous les agents préfèrent une alternative à une autre, alors la relation de préférence sociale vérifie aussi cette préférence.
- *Transitivité* : la relation de préférence sociale est complète et transitive.
- *Indépendance des alternatives non-pertinentes* : le classement relatif de deux alternatives ne dépend que de leur position relative dans les préférences individuelles (et non de la position d'autres alternatives) ; autrement dit, si nous restreignons les préférences individuelles à un sous-ensemble d'alternatives et que nous appliquons la règle de vote sur ce nouveau profil, alors les positions relatives de deux alternatives de ce sous-ensemble sont les mêmes que celles obtenues en considérant tout l'ensemble.
- *Absence de dictateur* : il n'existe aucun agent pour lequel le classement personnel coïncide avec la préférence sociale, indépendamment des préférences des autres.

À première vue, il semble naturel voire nécessaire qu'une règle de vote vérifie toutes ces propriétés. Malheureusement, en considérant aussi l'axiome *universalité* (cf Section 1.2.1), le système d'axiomes

obtenu devient beaucoup trop contraignant :

Théorème 3 (Arrow [1951])**.** *Dans le cas où l'ensemble des alternatives possibles contient au moins trois éléments, il n'existe pas de fonction d'agrégation des préférences vérifiant à la fois les axiomes* universalité*,* unanimité*,* transitivité*,* indépendance des alternatives non-pertinentes *et* absence de dictateur*.*

Soulignons toutefois que ce théorème d'impossibilité ne s'applique pas aux systèmes de vote utilisant une information plus riche que des utilités ordinales. Par exemple, les systèmes recourant à une utilité cardinale par agent ne sont pas concernés par ce résultat, mais en contre partie, ils requièrent davantage d'efforts de la part des agents pour pouvoir construire des représentations numériques comparables.

Le théorème d'impossibilité d'Arrow peut aussi se traduire le façon suivante : tout système de vote vérifiant les axiomes *universalité*, *unanimité*, *transitivité* et *indépendance des alternatives non-pertinentes* est nécessairement une dictature ; ainsi, il est nécessaire d'abandonner au moins un de ces quatre axiomes pour éviter de tomber dans un système dictatorial. La règle majoritaire est un exemple de méthodes d'agrégation qui permet, en abandonnant l'axiome *universalité*, de vérifier tous les autres axiomes du théorème, dont l'axiome *absence de dictateur*. En effet, nous avons vu dans la sous-section précédente que l'hypothèse de *single-peakedness* sur les préférences impliquait la transitivité de la relation de préférence sociale (cf. théorème 2). De plus, il est facile de voir que la règle majoritaire vérifie tous les autres axiomes. Plus précisément, l'axiome *unanimité* est trivialement satisfait car l'ensemble de tous les agents constitue une majorité. L'axiome *indépendance des alternatives non-pertinentes* est aussi vérifié puisque le classement relatif de deux alternatives ne dépend que du nombre d'agents préférant la première à la seconde. Enfin, remarquons qu'il ne peut y avoir de dictateur dès lors qu'il est nécessaire d'obtenir la majorité des voix (sauf peut-être dans certaines situations de vote à deux agents, mais utiliser la règle majoritaire n'a pas de sens dans ces situations).

Un autre résultat d'impossibilité très célèbre dans la littérature concerne la manipulabilité des fonctions de choix social. Ce résultat est connu sous le nom de théorème de Gibbard-Satterthwaite, du nom du professeur émérite de philosophie Allan Gibbard et de celui de l'économiste Mark Allen Satterthwaite. Pour pouvoir énoncer précisément ce résultat, nous avons besoin d'introduire les propriétés suivantes :

- *Contrainte de portée* : la fonction de choix social retourne au moins trois alternatives différentes sur le domaine des profils admissibles.
- *Non-manipulabilité* : il n'existe pas de profil admissible qui soit manipulable par un agent, où la manipulabilité est définie comme suit : un agent i peut transmettre une fausse relation de préférence $\succ_i$ de sorte que la fonction de choix social retourne une alternative qu'il préfère strictement à celle qui serait retournée s'il transmettait ses véritables préférences.

Avec ces propriétés, nous obtenons le résultat d'impossibilité suivant :

Théorème 4 ([Gibbard, 1973, Satterthwaite, 1975])**.** *Dans le cas où l'ensemble des alternatives possibles contient au moins trois éléments, il n'existe pas de fonction de choix social qui soit mono-évaluée et qui vérifie* universalité*,* absence de dictateur*,* contrainte de portée *et* non-manipulabilité*.*

En conséquence, toute fonction de choix social mono-évaluée et définie pour tout profil (axiome *universalité*) est soit dictatoriale soit manipulable, à moins que certaines alternatives ne peuvent jamais être déclarées gagnantes, qu'importe le profil de préférences considéré. Ainsi, nous constatons une nouvelle fois que la caractérisation d'une fonction de choix social "raisonnable" n'est pas un exercice élémentaire, et nécessite de faire un compromis entre les différentes propriétés désirées et celles qui seront engendrées par ces dernières. À titre d'exemple, en abandonnant l'axiome *universalité* pour se concentrer sur les profils *single-peaked* (cf. définition 23), la règle majoritaire vérifie tous les autres axiomes du théorème (e.g., Moulin [1980]).

1.2.3 Règles de vote étudiées

Dans cette sous-section, nous présentons les règles de vote que nous avons étudiées durant cette thèse, à savoir la méthode de Borda et le vote par approbation.

La méthode de Borda

Jean-Charles, chevalier de Borda, est un mathématicien et physicien français, contemporain du marquis de Condorcet, dont les contributions à la théorie du choix social sont souvent considérées comme faisant partie des plus importantes depuis les origines du domaine. Il s'est engagé dans de nombreux débats, notamment avec le marquis de Condorcet, sur les mérites respectifs des différents systèmes de vote, souvent considérés comme précurseurs de certains débats modernes sur la manière de répondre au théorème d'impossibilité d'Arrow. Il défendit en particulier un système de vote par scorage, aujourd'hui connu sous le nom de méthode de Borda. Cette méthode est actuellement utilisée à travers le monde, comme par exemple dans les élections parlementaires à Nauru ou encore durant de grandes compétitions sportives comme la ligue majeure de baseball aux États-Unis.

Une méthode de vote par scorage attribue à chaque alternative un certain nombre de points (ou score) dépendant de sa position dans les classements représentant les préférences de chaque agent. Plus précisément, plus une alternative est bien classée par un agent, plus celui-ci lui rapporte de points. Le score d'une alternative s'obtient en sommant tous les points qu'elle a recueilli auprès des différents agents. Les vainqueurs de l'élection sont alors les alternatives qui maximisent ce score. Notons que ces scores peuvent aussi servir à induire une relation de préférence sociale transitive, ce qui peut être utile lorsqu'un classement des candidats est requis. Avec la méthode de Borda, chaque agent procure à une alternative autant de points que d'alternatives qu'il considère comme étant plus mauvaises que celle-ci. Plus formellement, étant donné un profil de préférences $p = (\succ_1, \ldots, \succ_n)$, le score d'une alternative $x \in \mathcal{X}$ par la méthode de Borda est défini comme suit :

Définition 24 (score de Borda)**.** $\mathrm{B}(x, p) = \sum_{i \in N} |\{y \in \mathcal{X} : x \succ_i y\}|$.

La méthode Borda vérifie trivialement les axiomes *universalité*, *anonymat*, *neutralité* et *monotonie*. De ce fait, nous pouvons déduire du théorème de May (cf. théorème 1) que la méthode de Borda est équivalente à la règle majoritaire lorsque seulement deux alternatives sont à comparer. Pour d'autres

caractérisations de la méthode de Borda, le lecteur est invité à se référer aux travaux de Smith [1973], Young [1974] et Fishburn and Gehrlein [1976].

La méthode de Borda n'est pas Condorcet cohérente, comme toutes les méthodes de vote par scorage (e.g. [Moulin, 1991]) ; autrement dit, il n'est pas garanti que le vainqueur de Condorcet soit élu par la méthode de Borda lorsque celui-ci existe, comme le montre l'exemple suivant :

Exemple 7. *Reprenons l'exemple 6 dont le profil de préférences* $p = (\succ_1, \ldots, \succ_{1000})$ *est le suivant :*

$$\forall i \in \{1, \ldots, 499\},\ x^1 \succ_i x^2 \succ_i \ldots \succ_i x^m$$
$$\forall i \in \{500, \ldots, 1000\},\ x^m \succ_i x^2 \succ_i \ldots \succ_i x^1$$

Rappelons que l'alternative x^m *est le vainqueur de Condorcet de ce problème de décision. Les scores de Borda associés aux alternatives du problème sont les suivants :* $\mathrm{B}(x^1, p) = 499(m-1)$, $\mathrm{B}(x^m, p) = 501(m-1)$ *et* $\mathrm{B}(x^k, p) = 1000(m-k)$ *pour tout* $k \in \{2, \ldots, m-1\}$. *Comme* $m \geq 3$, *nous pouvons déduire que c'est l'alternative* x^2 *qui maximise le score de Borda ; ainsi, le vainqueur de Condorcet n'est pas sélectionné ici. Néanmoins, l'alternative* x^2 *constitue un choix moins contestable que le vainqueur de Condorcet, car elle est classée deuxième par tous les agents, alors que celui-ci est considéré comme le pire choix possible par une quasi-majorité.*

Cet exemple permet d'illustrer le fait que la méthode de Borda permet de favoriser le consensus social plutôt que la majorité.

À présent, situons la méthode de Borda par rapport aux deux théorèmes d'impossibilité énoncés dans la sous-section précédente, autrement dit le théorème d'Arrow (cf. théorème 3) et le théorème de Gibbard–Satterthwaite (cf. théorème 4). Rappelons que ce premier théorème nous a permis de conclure qu'il est nécessaire d'abandonner au moins un des quatre axiomes suivants pour éviter de produire une procédure dictatoriale : *universalité*, *unanimité*, *transitivité* et *indépendance des alternatives non-pertinentes*. La méthode de Borda vérifie trivialement les axiomes *universalité*, *unanimité* et *transitivité*, mais ne satisfait pas l'axiome *indépendance des alternatives non-pertinentes* comme le montre le contre-exemple suivant :

Exemple 8. *Poursuivons l'exemple 7. Dans cet exemple, l'alternative* x^m *est le vainqueur de Condorcet tandis que l'alternative* x^2 *est le vainqueur de Borda. Supposons maintenant que, mis à part les alternatives* x^2 *et* x^m, *toutes les autres alternatives se sont retirées de l'élection avant que les agents aient pris une décision. Dans cette situation, le profil de préférences* p *se simplifie en un profil* $p' = (\succ_1, \ldots, \succ_{1000})$ *défini comme suit :*

$$\forall i \in \{1, \ldots, 499\},\ x^2 \succ_i x^m$$
$$\forall i \in \{500, \ldots, 1000\},\ x^m \succ_i x^2$$

À présent, les scores de Borda sont les suivants : $\mathrm{B}(x^2, p') = 499$ *et* $\mathrm{B}(x^m, p') = 501$. *Au vu de ces scores, nous concluons que l'alternative* x^m *est le nouveau vainqueur pour la méthode Borda. Nous ob-*

servons ainsi une inversion de la préférence sociale (concernant les candidats x^2 et x^m) lorsque nous nous concentrons sur un sous-ensemble des alternatives, ce qui montre que la méthode de Borda ne vérifie pas l'axiome indépendance des alternatives non-pertinentes*.*

Ainsi, en abandonnant l'axiome *indépendance des alternatives non-pertinentes*, la méthode de Borda "échappe" au théorème d'impossibilité d'Arrow en vérifiant toutes les autres propriétés du théorème, et notamment l'axiome *absence de dictateur*. En réalité, la plupart des procédures de vote ne vérifient pas l'axiome *indépendance des alternatives non-pertinentes*, comme notamment toutes les méthodes de vote par scorage. Ce résultat plutôt positif sur la méthode de Borda a pour contrepartie la manipulabilité de cette procédure de vote. En effet, d'après le théorème d'impossibilité de Gibbard-Satterthwaite (cf. théorème 4), il n'existe pas de procédure de vote vérifiant à la fois les axiomes *universalité*, *absence de dictateur*, *contrainte de portée* et *non-manipulabilité*. La méthode de Borda vérifiant trivialement les trois premières propriétés, nous pouvons facilement déduire que cette procédure de vote est manipulable. Pour illustrer ce dernier fait, nous pouvons étudier l'exemple suivant :

Exemple 9. *Considérons un problème de décision où trois agents doivent choisir collectivement une alternative de l'ensemble $\mathcal{X} = \{x^1, x^2, x^3, x^4, x^5, x^6\}$. Supposons que le profil $p = (\succ_1, \succ_2, \succ_3)$ associé à ce problème est le suivant :*

$$\begin{array}{c} x^6 \succ_1 x^5 \succ_1 x^4 \succ_1 x^3 \succ_1 x^2 \succ_1 x^1 \\ x^1 \succ_2 x^2 \succ_2 x^3 \succ_2 x^4 \succ_2 x^5 \succ_2 x^6 \\ x^1 \succ_3 x^2 \succ_3 x^3 \succ_3 x^4 \succ_3 x^5 \succ_3 x^6 \end{array}$$

Observons que les agents 2 et 3 ont exactement la même relation de préférence sur les alternatives. Les scores de Borda associés à ce profil sont les suivants : $\mathrm{B}(x^1, p) = 10$, $\mathrm{B}(x^2, p) = 9$, $\mathrm{B}(x^3, p) = 8$, $\mathrm{B}(x^4, p) = 7$, $\mathrm{B}(x^5, p) = 6$ *et* $\mathrm{B}(x^6, p) = 5$. *De ces scores, nous déduisons que l'alternative x^1 est l'unique vainqueur par la méthode de Borda. Notons que l'agent 1 considère cette alternative comme étant le plus mauvais choix possible. Supposons un instant que ce dernier décide de transmettre au système la fausse relation de préférence $\succ_1'$ définie par :*

$$x^3 \succ_1' x^5 \succ_1' x^4 \succ_1' x^6 \succ_1' x^2 \succ_1' x^1$$

Avec le profil $p' = (\succ_1', \succ_2, \succ_3)$, nous obtenons les scores de Borda suivants : $\mathrm{B}(x^1, p') = 10$, $\mathrm{B}(x^2, p') = 9$, $\mathrm{B}(x^3, p') = 11$, $\mathrm{B}(x^5, p') = 6$ *et* $\mathrm{B}(x^6, p') = 2$. *Dans cette situation, c'est l'alternative x^3 qui serait déclarée vainqueur par la méthode de Borda. Ainsi, en connaissant les préférences des autres votant, l'agent 1 peut avoir envie de transmettre cette fausse relation de préférence pour voir gagner l'alternative x^3 qu'il préfère strictement à l'alternative x^1. Cette situation montre que la méthode de Borda est une règle de vote manipulable.*

Vote par approbation

Le vote par approbation, aussi appelé vote par assentiment, est un système de vote assez simple, étudié principalement depuis les années soixante-dix (e.g., [Ottowell, 1977, Weber, 1977, Kellett and Mott, 1977, Brams and Fishburn, 1978, Morin, 1980, Brams and Fishburn, 1983]) et utilisé actuellement par de nombreuses associations pour mettre en place diverses élections. Parmi ces associations, nous pouvons mentionner l'association mathématique d'Amérique, l'institut des ingénieurs électriciens et électroniciens ainsi que l'association américaine de statistique. Le vote par approbation est un système dit *non-rangé*, celui-ci n'ayant pas besoin de connaître la relation de préférence $\succ_i$ de chaque agent $i \in N$ pour pouvoir déterminer le gagnant de l'élection. Pour mettre en œuvre cette procédure de vote, il suffit à la place que chaque agent $i \in N$ transmette au système un ensemble $\mathcal{X}_i \subseteq \mathcal{X}$ représentant l'ensemble des alternatives qui ont son approbation ; la donnée $(\mathcal{X}_1, \ldots, \mathcal{X}_n)$ sera appelée *profil d'approbation* dans la suite de cette sous-section. Étant donné un tel profil, le vote par approbation déclare vainqueur l'alternative ayant obtenu le plus grand nombre d'approbations. Ce système de vote maximise donc un score, appelé *score d'approbation*, formellement défini comme suit pour toute alternative $x \in \mathcal{X}$:

Définition 25 (score d'approbation)**.** $\mathrm{A}(x, N) = |i \in N : x \in \mathcal{X}_i|$

Notons que si un agent décidait de transmettre au système un ensemble d'alternatives vide, alors son vote serait équivalent à une abstention. Pour simplifier la présentation, nous supposerons donc que les ensembles $\mathcal{X}_i$ sont tous non-vides dans la suite de cette sous-section.

Les scores d'approbation peuvent aussi être utilisés pour construire la relation de préférence sociale $\succsim$ définie par : $x \succsim y$ si et seulement si $\mathrm{A}(x, N) \geq \mathrm{A}(y, N)$. Comme cette relation de préférence est transitive par définition, alors le vote par approbation satisfait l'axiome de *transitivité* introduit en Section 1.2.2. Par ailleurs, le vote par approbation vérifie trivialement tous les axiomes du théorème de May (cf. théorème 1) adaptés aux profils d'approbation. Néanmoins, ces derniers ne suffisent pas à caractériser le vote par approbation pour la décision collective fondée sur les profils d'approbation. En effet, la procédure de vote maximisant le score $s(x, N) = \sum_{i \in N : x \in \mathcal{X}_i} 1/|\mathcal{X}_i|$ satisfait aussi tous ces axiomes. Cependant, en considérant un axiome d'anonymat plus exigeant que celui du théorème de May, nous pouvons obtenir une caractérisation du vote par approbation, et ce qu'importe le nombre d'alternatives du problème (cf. [Goodin and List, 2006] pour une présentation de ce nouvel axiome). Pour d'autres axiomatisations, le lecteur est invité à consulter les travaux de Fishburn [1978] et Sertel [1988].

Lorsque le vainqueur de Condorcet existe, le vote par approbation est susceptible de l'élire (e.g., [Baujard et al., 2014]), mais ce n'est pas toujours le cas, comme le montre l'exemple suivant :

Exemple 10. *Reprenons le problème de vote introduit dans l'exemple 6 dont le profil de préférences est le suivant :*

$$\forall i \in \{1, \ldots, 499\},\ x^1 \succ_i x^2 \succ_i \ldots \succ_i x^m$$
$$\forall i \in \{500, \ldots, 1000\},\ x^m \succ_i x^2 \succ_i \ldots \succ_i x^1$$

Supposons que les agents approuvent uniquement les deux premières alternatives de leur classement individuel. Dans cette situation, le système observe le profil d'approbation suivant :

$$\forall i \in \{1, \ldots, 499\}, \mathcal{X}_i = \{x^1, x^2\}$$
$$\forall i \in \{500, \ldots, 1000\}, \mathcal{X}_i = \{x^m, x^2\}$$

Nous avons alors $\mathrm{A}(x^1, N) = 499$, $\mathrm{A}(x^2, N) = 1000$, $\mathrm{A}(x^m, N) = 501$ *et* $\mathrm{A}(x^k, N) = 0$ *pour tout* $k \in \{3, \ldots, m-1\}$. *C'est donc l'alternative* x^2 *qui est déclarée vainqueur par le vote par approbation, et non* x^m *le vainqueur de Condorcet. Toutefois, comme nous l'avons déjà remarqué, l'alternative* x^2 *est un choix moins contestable que le vainqueur de Condorcet dans ce problème, car ce dernier est considéré comme la pire alternative possible par près de la moitié des votants.*

De manière générale, en autorisant les agents à soutenir plusieurs alternatives à la fois de manière équivalente, le vote par approbation a tendance à choisir l'alternative la plus acceptable dans l'ensemble, et non celle considérée comme la meilleure par la majorité. Les primaires républicaines de 2016 aux États-Unis permettent d'illustrer ce dernier fait sur un cas réel : des sondages ont montré que Donald Trump n'était pas considéré comme un candidat acceptable par une portion importante des votants républicains, et donc qu'il n'aurait pas aussi bien réussi avec le vote par approbation [Brams, 2016].

Ce système de vote n'est généralement pas considéré comme étant une fonction de choix social à part entière, car celles-ci se fondent majoritairement sur des profils de préférences de type $(\succ_1, \ldots, \succ_n)$ au lieu des profils d'approbation de type $(\mathcal{X}_1, \ldots, \mathcal{X}_n)$. C'est d'ailleurs cette spécificité qui lui permet de ne pas tomber dans le théorème d'impossibilité d'Arrow (cf. théorème 3), le vote par approbation vérifiant trivialement tous les axiomes de ce théorème adaptés aux profils d'approbations. Pour le cas particulier de l'axiome *indépendance des alternatives non-pertinentes*, il convient toutefois de préciser que celui-ci est satisfait uniquement si les agents évaluent la qualité d'une alternative individuellement et indépendamment des autres alternatives du problème, en utilisant une échelle d'évaluation absolue qui leur est propre ; avec cette hypothèse, il est facile de voir que le vote par approbation vérifie cet axiome car l'ajout ou le retrait d'alternatives ne peut changer le score d'une alternative.

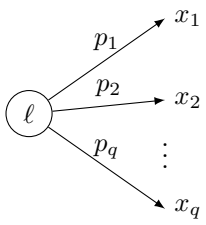
Bien que le théorème d'impossibilité Gibbard–Satterthwaite (cf. théorème 4) ne s'applique qu'aux fonctions de choix social, le vote par approbation se révèle sensible à la manipulation dans certaines situations. Tout d'abord, considérons le cas où les préférences de chaque agent sont représentables par une relation de préférence transitive (stricte ou non). Le bulletin de vote d'un agent est alors dit *sincère* si et seulement si toute alternative déclarée comme approuvée dans ce bulletin est préférée à toutes celles non approuvées. Par exemple, un agent i avec les préférences $x \succ_i y \succ_i z \succsim_i w$ est associé à cinq bulletins de vote sincères, qui sont les suivants : $\{x\}, \{x, y\}$, $\{x, y, z\}$, $\{x, y, w\}$ et $\{x, y, z, w\}$. Il a été récemment montré que, sous certaines hypothèses sur les préférences individuelles (e.g., agents "optimistes" ou maximisant une utilité espérée), aucune personne n'a vraiment intérêt à transmettre un bulletin de vote non sincère, car au moins un bulletin de vote sincère lui permettrait d'obtenir le même résultat, voire même de faire mieux (e.g. [Endriss, 2007]). Néanmoins, la coexistence de plusieurs bulletins

sincères peut en soi inciter les agents à voter stratégiquement, comme déjà souligné dans la littérature (e.g., [Niemi, 1984]), mais globalement, le vote par approbation est la méthode de vote non-rangée la moins manipulable (e.g., [Fishburn and Brams, 1981]). Considérons à présent le cas simple où les agents ont des préférences dichotomiques : chaque agent estime que certaines alternatives sont acceptables, ce qui divise l'ensemble des alternatives en deux catégories distinctes (les approuvées et les non approuvées), et chaque agent est indifférent entre toutes les alternatives appartenant à la même catégorie. Sous cette hypothèse, il existe un unique bulletin de vote sincère par agent, qui est tout simplement constitué de l'ensemble des alternatives que l'agent approuve. De plus, en présence de préférences dichotomiques, le vote par approbation est une procédure non-manipulable, et c'est d'ailleurs la seule méthode non-rangée ayant cette propriété [Brams and Fishburn, 1983].

1.3 Décision dans le risque

La décision dans le risque est un domaine de recherche qui s'intéresse aux problèmes de décision en environnement incertain. En guise d'illustration, un médecin devant prescrire un traitement pour un malade se retrouve face à un problème de décision dans l'incertain lorsque les traitements disponibles peuvent réussir comme échouer, voire provoquer de graves effets secondaires. On peut aussi penser à un opérateur de marché (aussi appelé *trader*) cherchant à dégager des gains sur des opérations d'achat et de vente, en tenant compte de l'incertitude des marchés financiers.

Dans son livre *"Risk, Uncertainty and Profit"*, Franck Knight regroupe sous le terme de *risque* toutes les situations décisionnelles telles que la probabilité de réalisation de chaque événement est précisément connue, alors que l'*incertain* est décrit comme un risque non mesurable [Knight, 1921]. Dans la littérature, différentes modélisations de l'incertitude ont été proposées, comme les mesures de possibilité et de nécessité (e.g., [Zadeh, 1978, Dubois and Prade, 1988]), les fonctions de croyance (e.g., [Dempster, 1967, Shafer, 1976]) ou encore les modèles à base de probabilités subjectives (e.g., [De Finetti, 1937, Savage, 1954, Schmeidler, 1986]). Dans le cadre de cette thèse, nous nous intéressons uniquement aux problèmes de décision dans le risque. Autrement dit, dans les problèmes considérés, chaque alternative possible est associée à une distribution de probabilités que le décideur connaît, définie sur un ensemble fini de conséquences. Sans perte de généralité, nous ferons l'hypothèse que les conséquences sont à valeurs réelles, représentant par exemple une somme d'argent (un gain si la valeur est positive, une perte sinon). Dans la théorie du risque, une telle distribution de probabilités est appelée une *loterie*. Nous noterons $\ell = (x_1, p_1; \ldots; x_q, p_q)$ la loterie associant à la conséquence x_j la probabilité $p_j > 0$ (avec $\sum_{j=1}^{q} p_j = 1$), et nous utiliserons la représentation graphique suivante :



Face à un problème de décision dans le risque, il s'agit de comparer les loteries associées aux alternatives possibles pour pouvoir déterminer la meilleure décision pour le décideur ; l'ensemble des loteries possibles sera noté $\mathcal{L} = \{\ell_1, \ldots, \ell_m\}$ dans la suite. Un exemple simple de problème de décision dans le risque est donné ci-dessous :

Exemple 11. *Une personne (le décideur) participe à un jeu qui peut lui permettre de gagner de l'argent. Cette personne peut soit décider de tirer une boule dans l'urne qui lui est présentée (décision d_1), soit décider de jouer à pile ou face (décision d_2). Dans le premier cas, selon la couleur de la boule tirée, il remporte les gains suivants :* 30 *euros pour une boule noire,* 20 *euros pour une boule rouge et* 5 *euros pour une boule verte. L'urne contient exactement* 1 *boule noire,* 2 *boules rouges et* 7 *boules vertes. Si le décideur choisit à la place de jouer à pile ou face, il gagne* 10 *euros s'il a parié sur le bon côté de la pièce, et rien sinon. Ainsi, les décisions d_1 et d_2 sont respectivement associées aux loteries suivantes :*

Finalement, pour pouvoir déterminer la meilleure décision entre tirer une boule et jouer à pile ou face, le décideur se retrouve à comparer les loteries de l'ensemble $\mathcal{L} = \{\ell_1, \ell_2\}$, où $\ell_1 = (30, 0.1; 20, 0.2; 5, 0.7)$ et $\ell_2 = (10, 0.5; 0, 0.5)$.

Pour aider le décideur dans sa prise de décision, il peut sembler naturel de comparer les différentes loteries possibles sur la base de l'espérance de gain. Formellement, l'espérance de gain d'une loterie $\ell = (x_1, p_1; \ldots; x_q, p_q)$ est défini comme suit :

Définition 26 (espérance de gain)**.** $E(\ell) = \sum_{j=1}^{q} p_j x_j$.

Dans l'exemple 11, nous avons $E(\ell_1) = 10.5 \geq E(\ell_2) = 5$, ce qui signifie que tirer une boule dans l'urne est la meilleure décision possible au sens de l'espérance de gain. Bien que ce critère paraisse raisonnable à première vue, les préférences observées ne sont généralement pas en accord avec ce que dicte l'espérance de gain, comme illustré par le célèbre paradoxe de Saint-Pétersbourg :

Exemple 12 (paradoxe de Saint-Pétersbourg)**.** *On propose au décideur de jouer à un jeu. Il s'agit de tirer plusieurs fois à pile ou face, tant que le côté "Face" n'est pas apparu. Si le côté " Face" a été obtenu au k-ième lancé, alors le décideur gagne 2^k euros. Nous cherchons à savoir combien le décideur est prêt à payer pour avoir le droit de participer à ce jeu. Supposons que les préférences du décideur s'expliquent par l'espérance de gain. Sous cette hypothèse, pour que le décideur ait envie de jouer, il suffit que sa mise soit plus petite que l'espérance de gain associée à ce jeu. Sachant que la probabilité que le premier "Face" apparaisse au k-ième lancé est égale à $1/2^k$, l'espérance de gain de ce jeu vaut exactement $\sum_{k=1}^{+\infty} 1/2^k \times 2^k = \sum_{k=1}^{+\infty} 1 = +\infty$. Ainsi, le décideur serait prêt à miser n'importe quelle*

somme d'argent pour pouvoir participer à ce jeu à espérance de gain infinie. Ce comportement ne semble pas tout à fait réaliste, car généralement, une personne n'est pas prête à miser plus que quelques euros pour avoir le droit de participer à ce jeu fondé sur pile ou face.

Ce paradoxe illustre le fait que l'espérance de gain ne permet généralement pas de modéliser les préférences du décideur dans un problème de décision dans le risque.

Dans cette section, nous nous intéressons à la modélisation des préférences en décision dans le risque, en commençant par présenter la *dominance stochastique du premier ordre* qui est le pendant dans l'incertain de la dominance de Pareto dans les problèmes de décision multicritère. Nous présenterons ensuite la *dominance stochastique du second ordre* qui permet de comparer le caractère risqué de deux loteries quelconques. Enfin, nous terminerons par une présentation du modèle de l'*utilité espérée*, qui permet notamment de corriger les défauts de l'espérance de gain mis en avant par le paradoxe de Saint-Pétersbourg.

1.3.1 Dominances stochastiques

La dominance stochastique du premier ordre permet, sans aucune connaissance sur les préférences du décideur, de détecter des loteries que tout agent rationnel considérerait comme étant sous-optimales par comparaison avec d'autres loteries du problème. Afin de définir formellement cette dominance, nous commençons par introduire la fonction décumulative associée à une loterie :

Définition 27 (fonction décumulative)**.** *La fonction décumulative* $G_\ell : \mathbb{R} \to [0,1]$ *associée à une loterie* $\ell = (x_1, p_1; \ldots; x_q, p_q)$ *est définie par :* $G_\ell(x) = \sum_{\substack{j \in \{1,\ldots,q\} \\ x \geq x_j}} p_j$.

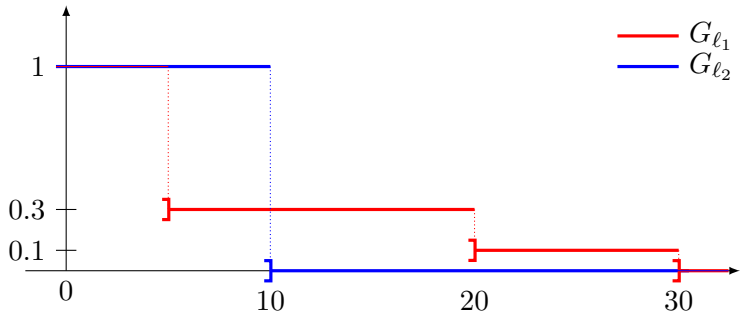
En d'autres termes, la fonction décumulative d'une loterie ℓ associe à toute valeur x la probabilité de gagner au moins x avec la loterie ℓ. À l'aide de cette définition, nous pouvons définir la dominance stochastique du premier ordre de la manière suivante :

Définition 28 (dominance stochastique du premier ordre)**.** *La loterie* $\ell \in \mathcal{L}$ *domine stochastiquement au premier ordre la loterie* $\ell' \in \mathcal{L}$*, noté* $\ell \succ_S^1 \ell'$*, si et seulement si :*

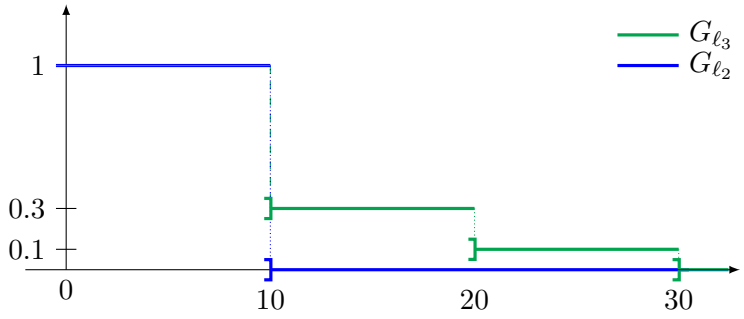
$$\forall x \in \mathbb{R},\ G_\ell(x) \geq G_{\ell'}(x) \text{ et } \exists x \in \mathbb{R}, G_\ell(x) > G_{\ell'}(x)$$

Par définition, si nous avons $\ell \succ_S^1 \ell'$, alors nos chances de gagner au moins un certain gain x sont plus grandes avec la loterie ℓ qu'avec la loterie ℓ', qu'importe le gain x considéré. Par conséquent, il paraît raisonnable de penser que le décideur préfère forcément la loterie ℓ à loterie ℓ' dès lors que nous avons $\ell \succ_S^1 \ell'$. Graphiquement, nous avons $\ell \succ_S^1 \ell'$ si et seulement si la courbe représentant la fonction décumulative de ℓ est au toujours au dessus de celle associée à ℓ' et que ces deux fonctions ne sont pas identiques. L'exemple ci-après permet d'illustrer graphiquement cette notion de dominance :

Exemple 13. *Reprenons la situation décisionnelle de l'exemple 11 où le décideur doit choisir entre les loteries* $\ell_1 = (30, 0.1; 20, 0.2; 5, 0.7)$ *et* $\ell_2 = (10, 0.5; 0, 0.5)$*. Les fonctions décumulatives associées à ces loteries sont représentées ci-dessous :*



De ce graphique, nous déduisons qu'aucune de ces loteries ne domine l'autre au sens de la dominance stochastique du premier ordre. En effet, nous avons $G_{\ell_1}(x) < G_{\ell_2}(x)$ *sur l'intervalle* $]5, 10]$*, mais* $G_{\ell_1}(x) > G_{\ell_2}(x)$ *sur l'intervalle* $]10, 30]$*. Par conséquent, nous avons besoin de plus d'informations sur les préférences du décideur pour pouvoir déterminer la décision la plus adaptée. À présent, changeons légèrement le jeu en donnant au décideur* 10 *euros (au lieu de* 5 *précédemment) si celui-ci tire une boule verte. La loterie* ℓ_1 *est alors remplacée par la loterie* $\ell_3 = (30, 0.1; 20, 0.2; 10, 0.7)$*, ce qui nous donne :*



Sur ce graphique, nous voyons que la loterie ℓ_3 *domine stochastiquement au premier ordre la loterie* ℓ_2*. Dans cette situation, nous pouvons conclure que la loterie* ℓ_3 *correspond à la meilleure décision possible pour le décideur et ce sans avoir besoin de recueillir des informations sur ses préférences subjectives.*

Bien que la dominance stochastique du premier ordre permette d'éliminer a priori des alternatives du problème, le nombre de loteries non stochastiquement dominées est potentiellement très grand (comme les solutions Pareto optimales en décision multicritère). Il s'agit alors d'enrichir la relation de préférence partielle induite par la dominance stochastique du premier ordre avec des informations sur le comportement du décideur vis-à-vis du risque. Afin d'estimer le caractère risqué d'une loterie, nous pouvons nous intéresser à la dispersion de la variable aléatoire représentant cette loterie. Par exemple, une loterie certaine, associant une conséquence avec une probabilité de 1, correspond à une variable aléatoire sans dispersion. Toute mesure de la dispersion d'une variable aléatoire est un candidat potentiel à la mesure du caractère risqué d'une loterie. Bien que le candidat le plus naturel pour un statisticien soit la variance, celle-ci ne permet généralement pas de modéliser un décideur ayant de l'aversion pour le risque. En effet, à espérance de gain égale, un décideur ayant de l'aversion pour le risque ne préfère pas toujours la loterie de plus petite variance (e.g., [Ingersoll, 1987]). En 1970, les économistes américains Michael Rotschild et Joseph Eugene Stiglitz ont proposé de comparer le caractère risqué de loteries en

utilisant le concept d'*accroissement de risque à moyenne constante* (*mean-preserving spread* en anglais) [Rothschild and Stiglitz, 1970]. Pour pouvoir définir formellement cette notion, nous commençons par introduire la dominance stochastique du second ordre :

Définition 29 (dominance stochastique du second ordre)**.** *La loterie $\ell \in \mathcal{L}$ domine stochastiquement au second ordre une loterie $\ell' \in \mathcal{L}$, noté $\ell \succ_S^2 \ell'$, si et seulement si :*

$$\forall x \in \mathbb{R}, \int_x^{+\infty} G_\ell(y)\,\mathrm{d}y \geq \int_x^{+\infty} G_{\ell'}(y)\,\mathrm{d}y \;\; et \;\; \exists x \in \mathbb{R}, \int_x^{+\infty} G_\ell(y)\,\mathrm{d}y > \int_x^{+\infty} G_{\ell'}(y)\,\mathrm{d}y$$

Par définition, la dominance stochastique du premier ordre implique la dominance stochastique du second ordre, mais l'implication inverse n'est pas toujours vérifiée. Avec cette définition, nous pouvons définir le concept d'*accroissement de risque à moyenne constante* de la manière suivante :

Définition 30 (accroissement de risque à moyenne constante)**.** *Une loterie $\ell' \in \mathcal{L}$ est un* accroissement de risque à moyenne constante *d'une loterie $\ell \in \mathcal{L}$ si et seulement si $E(\ell) = E(\ell')$ et $\ell \succ_S^2 \ell'$, où E désigne l'espérance de gain.*

Si la loterie ℓ' est un *accroissement de risque à moyenne constante* de la loterie ℓ, alors cette dernière loterie est considérée comme étant la moins risquée des deux. En distinguant le cas particulier où l'une des deux loteries est une loterie certaine, nous obtenons les deux indicateurs de comportement vis-à-vis du risque suivants :

Définition 31 (aversion faible au risque)**.** *Un décideur est dit* faiblement adversaire du risque *si et seulement s'il préfère toujours l'espérance d'une loterie à celle-ci.*

Définition 32 (aversion forte au risque [Cohen et al., 1987])**.** *Un décideur est dit* fortement adversaire du risque *si et seulement si celui-ci préfère toujours une loterie à une autre dès lors que cette dernière est un* accroissement de risque à moyenne constante *de la première.*

L'aversion forte au risque implique l'aversion faible au risque mais l'implication inverse n'est pas toujours vraie. De manière similaire, nous pouvons définir la notion d'attirance (faible ou forte) pour le risque : il suffit de considérer les préférences inverses de celles d'un décideur adversaire du risque. La notion de neutralité vis-à-vis du risque est quant à elle caractérisée par des indifférences. Ainsi, la notion d'*accroissement de risque à moyenne constante* permet de déduire des informations sur les préférences de tout agent rationnel concernant uniquement des loteries ayant la même espérance de gain.

Finalement, nous avons vu que les dominances stochastiques permettent de déduire des informations sur les préférences du décideur, mais que celles-ci ne sont généralement pas suffisantes pour pouvoir comparer toute paire de loteries et déterminer la meilleure décision pour le décideur.

1.3.2 Utilité espérée

Dans cette sous-section, nous nous intéressons à la modélisation des préférences par le modèle de l'*utilité espérée* [von Neumann and Morgenstern, 1947], aussi appelé EU dans la littérature (pour *Expected Utility* en anglais). Ce modèle décisionnel est en réalité celui qui est le plus utilisé en pratique lors

de prises de décision dans le risque. Cet engouement est notamment dû au fait que ce modèle est relativement simple à appréhender et possède des propriétés mathématiques qui facilitent son utilisation sur domaine combinatoire, comme par exemple dans le cadre de problèmes de décision séquentielle modélisés par des arbres de décision (e.g., [Raiffa, 1968]). Par ailleurs, nous allons voir que ce modèle décisionnel est soutenu par une justification axiomatique relativement élégante qui permet de motiver son utilisation dans des cas pratiques avec des arguments théoriques.

Les paramètres de ce modèle décisionnel sont sous la forme d'une fonction u à valeurs réelles, qui à toute conséquence $x \in \mathbb{R}$ associe l'utilité $u(x)$ de la conséquence x aux yeux du décideur. À partir d'une fonction u donnée, l'utilité espérée d'une loterie $\ell = (x_1, p_1; \ldots; x_q, p_q)$ est définie formellement de la manière suivante :

Définition 33 (utilité espérée)**.** $\mathrm{EU}(\ell, u) = \sum_{j=1}^{q} p_j u(x_j)$.

Ce modèle décisionnel est complètement caractérisé par un ensemble d'axiomes, relativement raisonnables, que nous présentons ci-après les uns après les autres.

Axiome de préordre total. La relation binaire $\succsim$ représentant les préférences du décideur est un préordre total ; autrement dit, celle-ci vérifie les propriétés suivantes :

- *Réflexivité* : $\ell \succsim \ell$ pour toute loterie ℓ,
- *Transitivité* : pour toutes loteries ℓ, ℓ' et ℓ'', si $\ell \succsim \ell'$ et $\ell' \succsim \ell''$, alors $\ell \succsim \ell''$.
- *Totalité* : pour toutes loteries ℓ et ℓ', nous avons soit $\ell \succsim \ell'$ soit $\ell' \succsim \ell$.

En d'autres termes, cet axiome stipule que le décideur est capable de comparer toute paire de loteries et de spécifier sa préférence, de sorte à pouvoir classer les loteries par ordre de préférences.

Pour pouvoir présenter les autres axiomes, nous avons besoin de définir l'opération de composition de loteries : pour toutes loteries ℓ', ℓ'' et toute valeur $\alpha \in [0, 1]$, la loterie composée $\ell = \alpha\ell' + (1-\alpha)\ell''$ associe la loterie ℓ' avec une probabilité α et la loterie ℓ'' avec la probabilité $1 - \alpha$. Pour le modèle EU, ainsi que pour la plupart des modèles de décision dans le risque, cette loterie en deux phases est équivalente à la loterie simple suivante : la probabilité de chaque conséquence possible est égale à la somme de sa probabilité dans ℓ' multipliée par la valeur α et de sa probabilité dans ℓ'' multipliée par $1 - \alpha$. Cette propriété est souvent appelée *axiome de réduction des loteries composées*.

Axiome de continuité. La relation binaire $\succsim$ représentant les préférences du décideur satisfait l'axiome de continuité si et seulement si pour toutes loteries ℓ, ℓ' et ℓ'' telles que $\ell \succsim \ell' \succsim \ell''$, il existe un couple $(\alpha, \beta) \in]0, 1[^2$ vérifiant :

$$\alpha\ell + (1-\alpha)\ell'' \succsim \ell' \succsim \beta\ell + (1-\beta)\ell''$$

Cet axiome est aussi connu sous le nom d'axiome Archimédien. Pour mieux nous rendre compte du fait que cet axiome assure la continuité de la relation de préférence, nous pouvons le traduire de la manière suivante : il existe toujours une valeur α suffisamment proche de 1 pour que la préférence $\ell \succsim \ell'$

ne soit pas inversée après avoir changé la loterie ℓ en la loterie composée $\alpha\ell + (1-\alpha)\ell''$. De plus , il existe toujours une valeur β suffisamment proche de 0 pour que la préférence $\ell' \succsim \ell''$ ne soit pas inversée après avoir changé la loterie ℓ'' en la loterie composée $\beta\ell + (1-\beta)\ell''$.

Axiome d'indépendance. La relation $\succsim$ représentant les préférences du décideur satisfait l'axiome d'indépendance si et seulement si, pour toutes loteries ℓ, ℓ' et ℓ'' et toute valeur $\alpha \in]0,1[$, nous avons :

$$\ell \succsim \ell' \;\Leftrightarrow\; \alpha\ell + (1-\alpha)\ell'' \succsim \alpha\ell' + (1-\alpha)\ell''$$

Autrement dit, la préférence $\ell \succsim \ell'$ ne doit pas s'inverser après avoir composé ces deux dernières loteries de la même manière avec une loterie quelconque (ℓ'' dans l'énoncé). Finalement, nous obtenons la caractérisation suivante :

Théorème 5 ([von Neumann and Morgenstern, 1947])**.** *Les deux propositions suivantes sont équivalentes :*

1. *La relation $\succsim$ vérifie les axiomes de préordre total, de continuité et d'indépendance.*
2. *Il existe une fonction u à valeurs réelles, unique à une transformation affine croissante près, vérifiant pour toutes loteries ℓ et ℓ' : $\ell \succsim \ell' \Leftrightarrow \mathrm{EU}(\ell,u) \geq \mathrm{EU}(\ell',u)$.*

Dans le cadre du modèle EU, la définition 31 concernant l'aversion faible au risque se traduit de la manière suivante : un décideur est faiblement adversaire du risque si et seulement si pour toute loterie $\ell = (x_1, p_1; \ldots; x_q, p_q)$, nous avons $\mathrm{EU}(\ell', u) \geq \mathrm{EU}(\ell, u)$ où $\ell' = (E(\ell), 1)$ est la loterie certaine qui associe l'espérance de gain de la loterie ℓ avec une probabilité de 1. Comme cette inégalité se réécrit $u(\sum_{j=1}^{q} x_j p_j) \geq \sum_{j=1}^{q} u(x_j)p_j$ (cf. définition 33), alors nous concluons qu'un décideur est faiblement adversaire du risque si et seulement si la fonction d'utilité u modélisant ses préférences est concave. Similairement, nous pouvons montrer qu'un décideur est attiré par le risque (resp. neutre vis-à-vis du risque) si et seulement si sa fonction d'utilité u est convexe (resp. linéaire).

Par ailleurs, avec une fonction u concave, la relation de préférence $\succsim$ induite par le modèle EU vérifie forcément $\mathrm{EU}(\ell, u) \geq \mathrm{EU}(\ell', u)$ pour toutes loteries ℓ et ℓ' telles que la loterie ℓ' est un *accroissement de risque à moyenne constante* de la loterie ℓ. De ce fait, en utilisant la définition 32 sur l'aversion forte au risque, nous concluons qu'un décideur avec une fonction u concave est fortement adversaire du risque ; autrement dit, un décideur faiblement adversaire du risque est aussi fortement adversaire du risque avec le modèle EU. Comme de plus l'aversion forte implique toujours l'aversion faible, alors ces deux notions de risque ne sont pas différenciées au sein du modèle EU. Ceci constitue une des limites descriptives bien connues de ce modèle décisionnel : le modèle EU ne permet pas de modéliser les préférences d'un individu hostile à l'introduction de risque dans une situation non risquée (décideur faiblement adversaire du risque) mais indifférent à l'accroissement du risque dans une situation qui est déjà risquée (décideur non fortement adversaire du risque). Le célèbre paradoxe d'Allais, du nom de l'économiste français Maurice Félix Charles Allais, est une illustration des limites descriptives du modèle EU :

Exemple 14 ([Allais, 1953])**.** *Considérons une expérience où les sujets sont interrogés individuellement sur leurs préférences concernant des loteries dont les conséquences sont monétaires (exprimées en mil-*

lions de francs). Dans un premier temps, il s'agit pour eux de spécifier la loterie qu'ils préfèrent parmi les deux loteries suivantes :

En pratique, il est souvent observé que la majorité des personnes interrogées préfèrent strictement la loterie ℓ_1 à la loterie ℓ'_1. En d'autres termes, les sujets préfèrent majoritairement gagner 100 *millions de francs de manière certaine, plutôt que tenter de remporter davantage au risque de ne rien gagner (et ce bien que la probabilité de ne rien gagner soit très faible). Ce phénomène, bien connu aujourd'hui, est souvent appelé* "effet de certitude"*. Avec le modèle* EU*, cette préférence se traduit ainsi :*

$$\begin{aligned} \ell_1 \succ \ell'_1 &\Leftrightarrow u(100) > 0.1u(500) + 0.89u(100) + 0.01u(0) \\ &\Leftrightarrow u(100) > \frac{0.1}{0.11}u(500) + \frac{0.01}{0.11}u(0) \end{aligned} \tag{1.4}$$

Les sujets de l'expérience sont ensuite interrogés sur leur préférence concernant les loteries suivantes :

Très souvent, la majorité des sujets dit préférer strictement la loterie ℓ'_2 à la loterie ℓ_2. En effet, comme la probabilité de gain est faible avec la loterie ℓ_2, les sujets sont souvent tentés par la possibilité de gagner cinq fois plus avec ℓ'_2 bien que la probabilité que cela se produise soit encore un petit peu plus faible. Avec le modèle EU*, ce comportement se traduit de la manière suivante :*

$$\begin{aligned} \ell'_2 \succ \ell_2 &\Leftrightarrow 0.1u(500) + 0.9u(0) > 0.11u(100) + 0.89u(0) \\ &\Leftrightarrow u(100) < \frac{0.1}{0.11}u(500) + \frac{0.01}{0.11}u(0) \end{aligned} \tag{1.5}$$

Cependant, il est facile de se rendre compte qu'aucune fonction d'utilité u ne vérifie les deux équations (1.4) et (1.5) en même temps, car celles-ci sont tout simplement contradictoires. Par conséquent, avec le modèle EU*, il n'est pas possible de modéliser les préférences $\ell_1 \succ \ell'_1$ et $\ell'_2 \succ \ell_2$ bien que ce point de vue soit partagé par la majorité des individus.*

Par ailleurs, l'axiome d'indépendance du modèle EU est souvent directement critiqué car, en pratique, il est fréquent que la majorité des individus interrogés ne respectent pas cet axiome. Afin d'illustrer ce fait, nous proposons d'étudier l'expérience suivante :

Exemple 15 ([Kahneman and Tversky, 1979])**.** *Une expérience a été menée sur des étudiants israéliens pour connaître leurs préférences sur des loteries dont les conséquences étaient exprimées dans la monnaie locale. Ces étudiants devaient tout d'abord comparer les loteries suivantes :*

Une grande majorité des étudiants ont alors répondu que la loterie ℓ_1 était meilleure que la loterie ℓ'_1 (effet de certitude). Puis, ils ont été interrogés sur les loteries suivantes :

Similairement à l'expérience du paradoxe d'Allais, nous observons ici une inversion de la préférence, avec des étudiants préférant majoritairement la loterie ℓ'_2 à la loterie ℓ_2. Remarquons que ces deux dernières loteries peuvent se réécrire de la manière suivante : $\ell_2 = 0.25\ell_1 + 0.75\ell_3$ et $\ell'_2 = 0.25\ell_2 + 0.75\ell_3$, où $\ell_3 = (0, 1)$ est la loterie associant la conséquence 0 avec une probabilité de 1. Par conséquent, d'après l'axiome d'indépendance, la préférence $\ell_1 \succ \ell'_1$ devrait induire la préférence $\ell_2 \succ \ell'_2$, ce qui n'est pas le cas ici. Autrement dit, les préférences de ces étudiants ne vérifient pas l'axiome d'indépendance et donc ne sont pas représentables par le modèle EU*.*

Ces limites descriptives du modèle EU justifient la présence d'autres modèles décisionnels pour la modélisation des préférences dans le risque. En particulier, nous pouvons citer les travaux de Machina [1982] sur l'*utilité espérée généralisée*, rejetant totalement l'axiome d'indépendance, ou encore les travaux de [Quiggin, 1992] sur l'*utilité espérée dépendant du rang* qui repose sur une restriction de cet axiome. Néanmoins, dans le cadre de cette thèse, nous avons uniquement étudié le modèle EU, les autres modèles faisant partie de nos perspectives de recherche.

1.4 Élicitation des préférences

Dans les sections précédentes, nous avons présenté différents modèles permettant de représenter les préférences d'individus dans des contextes de décision multicritère, multi-agents et dans le risque. Bien que la plupart de ces modèles se justifient très souvent par un système d'axiomes relativement désirables, ces derniers n'indiquent généralement pas les paramètres du modèle qu'il convient d'utiliser pour pouvoir représenter au mieux les préférences dans un contexte décisionnel donné. L'élicitation des préférences est une problématique de recherche dont l'objectif est de concevoir des systèmes d'aide à la décision qui, en recueillant des informations sur les préférences des agents, sont capables de produire des recommandations pertinentes et personnalisées. Un tel système doit être en mesure de modéliser les préférences des utilisateurs avec précision car la moindre inexactitude, aussi petite soit-elle, peut conduire à une recommandation radicalement différente, comme le montre l'exemple simple suivant :

Exemple 16. *Un décideur doit faire un choix entre les alternatives a et b dont les évaluations multi-critères sont les suivantes : $a = (10, 30)$ et $b = (20, 10)$. Supposons que les préférences du décideur soient représentables par une somme pondérée* SP *(cf. définition 11) dont le vecteur de poids $\omega = (\omega_1, \omega_2)$ reste à déterminer. Le décideur nous informe que le premier critère est approximativement deux fois plus important que le second critère. Considérons alors le vecteur de poids $\omega = (2/3 + \varepsilon, 1/3 - \varepsilon)$, où ε est une valeur arbitrairement proche de zéro. Avec ce vecteur de poids, nous obtenons $\mathrm{SP}(a, \omega) = 50/3 - 20\varepsilon$ et $\mathrm{SP}(b, \omega) = 50/3 + 10\varepsilon$. Les deux alternatives sont donc équivalentes si $\varepsilon = 0$. Cependant, pour tout $\varepsilon < 0$ arbitrairement proche de zéro, la solution a est strictement meilleure que la solution b, tandis que le contraire se produit avec $\varepsilon > 0$ arbitrairement proche de zéro. Ainsi, une petite variation du vecteur de poids ω conduit ici à une décision différente. Par conséquent, dans cette situation décisionnelle, il semble nécessaire de recueillir davantage d'informations sur les préférences du décideur avant de formuler une quelconque recommandation.*

Cet exemple illustre le fait que l'élicitation des paramètres d'un modèle décisionnel est une tâche délicate en soi, qui nécessite des méthodes automatiques permettant de produire une modélisation précise, en collectant des informations pertinentes sur les préférences du décideur. L'élicitation des préférences est en réalité une tâche cruciale dans de nombreuses applications comme le marketing personnalisé, le commerce électronique et les systèmes de recommandation (citons par exemple Deezer pour les musiques et Netflix pour les vidéos). Dans cette section, nous commençons par présenter brièvement les approches classiques pour la problématique de l'élicitation des préférences (cf. Section 1.4.1), avant de centrer la présentation sur l'élicitation incrémentale fondée sur le concept de Minimax Regret (cf. Section 1.4.2), cette dernière approche constituant l'angle d'attaque principal de cette thèse.

1.4.1 Un aperçu des approches standards

Dans cette sous-section, nous discutons brièvement des avantages et des inconvénients des approches standards pour la problématique de l'élicitation des préférences.

Élicitation totale d'une fonction d'utilité

En pratique, l'ensemble des alternatives possibles est souvent doté d'une structure multidimensionnelle, les alternatives étant généralement évaluées selon plusieurs points de vue simultanément. Dans ce cas, chaque alternative est associée à un point dans un espace multi-attributs. L'approche classique en élicitation des préférences vise à déterminer le modèle de préférences du décideur sur tout l'espace multi-attributs, point par point, à l'aide d'une série de questions-réponses (e.g., [Fishburn, 1967, Krantz et al., 1971, Keeney and Raiffa, 1976]). Ce modèle, après avoir été construit par des interactions avec le décideur, peut ensuite être utilisé pour formuler une recommandation personnalisée : l'alternative optimale au sens de ce modèle constitue la meilleure option pour le décideur. Une fois construit, ce modèle peut aussi servir à ordonner toutes les alternatives. Cette possibilité est particulièrement intéressante lorsque le décideur souhaite retenir plusieurs options ou bien lorsque celui-ci a besoin de produire un classement complet de

toutes les alternatives. Par ailleurs, le modèle obtenu peut aussi être exploité plus tard, pour résoudre de nouvelles instances du même problème de décision ; à titre d'exemple, on peut penser à l'ajout de nouveaux films dans la base de données d'un système de recommandation de vidéos.

Néanmoins, cette approche est difficile à mettre en œuvre sur domaine combinatoire, car celle-ci devient très rapidement coûteuse en nombre de questions. Par exemple, en décision multicritère, le nombre de solutions croît exponentiellement avec le nombre de critères considérés dans le problème. De ce fait, apprendre la fonction d'utilité du décideur point par point dans le cadre de la décision multicritère ne semble pas être une option envisageable en pratique (sauf peut-être pour des modèles décisionnels décomposables très simples). En effet, dans ces situations, il n'est pas raisonnable de penser que le décideur accepte de passer des journées entières à répondre à un questionnaire lui permettant de prendre une décision finale (à moins que l'enjeu de la décision soit d'une importance critique). Par ailleurs, cette approche n'est généralement pas à l'abri de réponses contradictoires ou biaisées (e.g., [Simon, 1955, Tversky and Kahneman, 1975, Camerer et al., 2003]), ce qui constitue une autre difficulté.

Optimisation interactive

L'optimisation interactive est une approche très largement étudiée en décision multicritère, qui permet de diriger l'exploration de l'ensemble des solutions Pareto-optimales en fonction des différentes interactions avec le décideur, sans jamais avoir à énumérer toutes ces solutions (e.g., [Zionts and Wallenius, 1976, Vanderpooten, 1989, Vanderpooten and Vincke, 1997, Miettinen, 1999, Korhonen, 2005, Greco et al., 2016]). Cette approche repose généralement sur l'utilisation d'une fonction d'agrégation paramétrée dont le rôle est de véhiculer les informations que le système a recueillies sur les préférences du décideur. Les fonctions d'agrégation majoritairement utilisées sont la somme pondérée, la norme de Tchebycheff ou encore des fonctions fondées sur un niveau d'aspiration (e.g., [Wierzbicki, 1986]). L'idée générale est d'alterner les deux étapes suivantes :

- *Calcul :* évaluation des solutions en utilisant une instance possible des paramètres préférentiels.
- *Dialogue :* présentation d'un ensemble de solutions au décideur, choisies soigneusement par le système, pour que le décideur lui transmette de nouvelles informations sur ses préférences.

Le processus d'élicitation s'arrête dès que le décideur rencontre une solution qui lui convient. Les différentes méthodes d'optimisation interactive se distinguent les unes des autres notamment par le type de préférences que le décideur peut exprimer. Par exemple, le décideur peut spécifier la solution qu'il préfère parmi un ensemble (e.g., [Zionts and Wallenius, 1976, Steuer, 1986]) ou encore préciser sur quels critères il est prêt à faire des concessions pour améliorer les performances réalisées sur d'autres critères (e.g., [Benayoun et al., 1971]).

Cette approche relativement générique permet de traiter tout problème d'optimisation combinatoire multicritère pourvu qu'il existe un algorithme de résolution efficace avec paramètres préférentiels précis. En effet, pour les autres problèmes, le temps d'attente entre chaque évaluation risque de conduire le décideur à interrompre les interactions. Par ailleurs, cette approche présente l'avantage de pouvoir s'accompagner d'une interface graphique (e.g., [Korhonen and Laakso, 1986, Lewandowski and Granat, 1991,

Klimberg, 1992]), permettant au décideur de visualiser les solutions possibles durant l'exploration et de le conforter (ou pas) dans ses choix. Cependant, les méthodes d'optimisation interactive ne permettent généralement pas de garantir que la solution finale constitue réellement la meilleure alternative possible pour le décideur. En effet, la valeur finale des paramètres ne représente pas nécessairement au mieux les préférences du décideur, ce dernier ayant pu choisir d'interrompre le processus d'élicitation par lassitude après avoir détecté une alternative qu'il estime relativement satisfaisante. Enfin, soulignons le fait que de nouvelles interactions avec le décideur peuvent être requises pour résoudre d'autres instances du même problème (par exemple, suite à l'ajout de nouvelles alternatives dans la base de recommandation).

Apprentissage à partir d'une base de données

Dans la littérature, on retrouve aussi des méthodes qui considèrent en entrée une base de données contenant des informations sur les préférences du décideur, et qui visent à déterminer les paramètres du modèle qui permettent d'expliquer au mieux cette base de données. Par exemple, la méthode UTA apprend une fonction d'utilité additive par programmation linéaire à partir d'une base de données contenant des informations ordinales (e.g., [Jacquet-Lagrèze and Siskos, 1982, Siskos and Yannacopoulos, 1985]). Ces méthodes peuvent se distinguer les unes des autres par la nature des données exploitées. À titre d'exemple, le décideur peut être amené à exprimer ses préférences sur un petit ensemble d'alternatives (fictives ou non), de préciser si certains critères sont plus importants que d'autres, voire même d'accompagner ces informations d'intensités de préférences. Ces données collectées sont ensuite utilisées pour contraindre le modèle décisionnel de sorte à respecter le plus possible ces préférences observées. La plupart de ces méthodes se présentent alors sous la forme de problèmes d'optimisation sous contraintes. Par exemple, il a été proposé de minimiser des erreurs d'utilité (e.g., [Jacquet-Lagrèze and Siskos, 1982, Sobrie et al., 2017a]) ou encore de minimiser un critère d'erreur quadratique (e.g., [Murofushi and Mori, 1989, Grabisch et al., 1995, Meyer and Roubens, 2006, 2005]) à la manière des méthodes classiques utilisées en apprentissage automatique. La fonction à optimiser peut aussi servir à discriminer entre les différents paramètres compatibles avec la base de données. En effet, nous pouvons par exemple chercher à maximiser les écarts de performance entre des solutions dont le classement relatif est connu (e.g., [Marichal and Roubens, 2000, Beuthe and Scannella, 2001, Grabisch et al., 2008]), à minimiser la variance du modèle pour être le moins spécifique possible (e.g., [Kojadinovic, 2007]) ou encore à minimiser une entropie relative pour tenir compte des erreurs éventuelles dans la base (e.g., [Bous and Pirlot, 2013]). Signalons que certains logiciels d'optimisation mettent actuellement en œuvre ces techniques d'élicitation (e.g., [Marichal et al., 2005, Huédé et al., 2006, Grabisch et al., 2008]).

L'intérêt principal de cette approche est que le processus d'élicitation ne dépend pas des alternatives du problème. Ceci permet notamment de résoudre des problèmes d'optimisation combinatoire après avoir déterminé les paramètres du modèle qu'il convient d'utiliser. Par ailleurs, une fois le modèle de préférence construit, celui-ci peut être exploité pour résoudre de nouvelles instances du même problème de décision, sans nécessiter d'interactions supplémentaires. Il est toutefois important de souligner que l'efficacité de l'approche est fortement impactée par la qualité de la base de données : plus celle-ci est riche, mieux les

préférences sont approchées. Cette observation suggère de construire le modèle de préférence après avoir collecté le plus de données possibles sur les préférences du décideur. Cependant, plus la taille de la base de données augmente, plus le risque de rencontrer des incohérences est élevé. De ce fait, un système de gestion des incohérences est très souvent indispensable en pratique pour pouvoir déterminer au moins une instance des paramètres compatible avec les préférences observées. De manière générale, obtenir une base de données de qualité est une tâche relativement difficile car le nombre d'interactions possibles avec le décideur est généralement limité. Une autre faiblesse de cette approche réside dans le choix de la fonction à optimiser (e.g., erreur quadratique, variance) qui semble parfois relativement arbitraire. Ce dernier fait a d'ailleurs conduit des chercheurs en aide à la décision à travailler avec toutes les instances des paramètres compatibles avec les informations contenues dans la base de données (e.g., [Greco et al., 2010c, Angilella et al., 2010, Greco et al., 2014]). Soulignons toutefois que la prise en compte simultanée de plusieurs instances peut entraîner des situations d'indécision, lorsque les recommandations changent en fonction de l'instance considérée.

Approche bayésienne

Lorsque l'incertitude sur la fonction d'utilité du décideur est quantifiée à l'aide d'une distribution de probabilité a priori, le critère de l'espérance d'utilité peut être utilisé pour évaluer les différentes alternatives du problème. Cette distribution de probabilité peut par exemple avoir été obtenue à partir de données disponibles sur des personnes ayant des goûts relativement similaires à ceux du décideur. Sans aucune autre information sur les préférences du décideur, il semble raisonnable de lui recommander l'alternative qui maximise l'espérance d'utilité a priori. Si en revanche des interactions avec le décideur sont possibles, nous pouvons envisager de lui poser des questions afin de mettre à jour cette distribution de probabilité en fonction de ses propres préférences ; une question est généralement considérée comme informative si, en espérance, la réponse du décideur conduit à une augmentation de l'espérance d'utilité maximale. Une stratégie d'élicitation possible consiste à poser progressivement des questions au décideur de manière à améliorer l'espérance d'utilité maximale jusqu'à satisfaire un certain critère d'arrêt (e.g., [Chajewska et al., 2000, Braziunas and Boutilier, 2006, Viappiani and Boutilier, 2010]). Afin de limiter le nombre d'interactions avec le décideur, il convient bien évidemment de choisir, à chaque itération, la question offrant la meilleure amélioration en espérance. L'évaluation d'une question devrait en théorie être réalisée de manière séquentielle, en considérant toutes les futures questions et réponses possibles (e.g., [Boutilier, 2002]). Cependant, pour réduire les temps de calcul, cette évaluation est souvent effectuée de façon "myope" en pratique, autrement dit en comparant uniquement la valeur de l'espérance d'utilité maximale avant et après avoir posé la question.

Dans de nombreux travaux, il a été observé que l'approche bayésienne permet de formuler une recommandation pertinente et personnalisée en posant relativement peu de questions au décideur en pratique (e.g., [Chajewska et al., 2000, Viappiani and Boutilier, 2010]). Par ailleurs, cette approche peut aussi être utilisée pour résoudre des problèmes d'optimisation combinatoire, à condition de pouvoir calculer efficacement l'espérance d'utilité maximale sur l'espace des solutions. Remarquons toutefois que

ces méthodes nécessitent de connaître la distribution de probabilité a priori sur les fonctions d'utilité, ce qui n'est pas toujours réalisable en pratique. En particulier, cela semble difficile dans le cadre d'une prise de décision imprévue ou spécifique au décideur. Enfin, une autre faiblesse de cette approche provient des critères de décision utilisés qui sont généralement difficiles à optimiser en soi (e.g., maximiser l'espérance d'utilité, minimiser l'espérance de perte) et qui sont par conséquent très souvent approximés en pratique (e.g., à l'aide de méthodes de Monte-Carlo).

1.4.2 Approche incrémentale

Dans l'optique de réduire l'effort d'élicitation, des procédés d'élicitation partielle ont été proposés, posant progressivement des questions au décideur, soigneusement choisies les unes après les autres, de sorte à restreindre l'espace des paramètres possibles jusqu'à être en mesure de formuler une recommandation. Cette approche dite "incrémentale" remonte à la méthode ISMAUT [White III et al., 1984] conçue pour les modèles décisionnels additifs, et s'est développée par la suite dans la communauté de l'intelligence artificielle, qui utilise notamment le critère de décision Minimax Regret pour déterminer les prochaines questions à poser et l'alternative à recommander (e.g., [Boutilier, 2002, Boutilier et al., 2006]). L'approche d'élicitation incrémentale est maintenant utilisée de manière standard pour résoudre de nombreux problèmes, comme l'élicitation d'utilités de von Neumann-Morgenstern (e.g., [Wang and Boutilier, 2003]), l'élicitation d'utilités multi-attributs (e.g., [White III et al., 1984, Braziunas and Boutilier, 2007]) ou encore l'élicitation des préférences individuelles dans un problème de vote (e.g., [Kalech et al., 2010, Lu and Boutilier, 2011b, Dery et al., 2014, 2016]). L'atout majeur de cette approche est de limiter l'effort du décideur en ne lui posant que des questions visant à discriminer entre les alternatives du problème, et non à préciser les paramètres qui représentent au mieux ses préférences. L'efficacité pratique de cette approche a été démontrée dans de nombreux travaux (e.g., [Wang and Boutilier, 2003, Braziunas, 2011]) et notamment durant une étude menée avec de vrais utilisateurs [Braziunas and Boutilier, 2010].

Dans cette sous-section, nous commençons par présenter brièvement la méthode ISMAUT avant de nous intéresser plus particulièrement aux méthodes incrémentales fondées sur le critère de décision Minimax Regret. En effet, le principal challenge de cette thèse est de parvenir à étendre efficacement cette dernière approche à des modèles décisionnels complexes et à des problèmes d'optimisation combinatoire.

La méthode ISMAUT

La méthode ISMAUT (pour *imprecisely specified multiattribute utility theory* en anglais) fait partie des premières méthodes incrémentales proposées pour la prise de décision en présence de préférences imprécisément définies [White III et al., 1984]. Comme son nom l'indique, celle-ci s'applique aux situations de décision dans lesquelles les alternatives possibles sont évaluées selon plusieurs points de vue ; notons $\mathcal{X}$ l'ensemble des alternatives possibles et q le nombre de points de vue considérés dans le problème. Dans ces situations, cette méthode suppose que les préférences du décideur sont représentables par une

fonction d'utilité additive u prenant la forme suivante (e.g., [Fishburn, 1968, Keeney and Raiffa, 1976]) :

$$u(x) = \sum_{j=1}^{q} \omega_j v_j(x_j)$$

où les paramètres $\omega = (\omega_1, \ldots, \omega_q)$ et $v = (v_1, \ldots, v_q)$ sont respectivement un vecteur de poids et un vecteur de fonctions de valeurs locales. Lorsque les paramètres de la fonction u ne sont pas connus de manière précise, nous pouvons considérer l'ensemble U de toutes les fonctions u compatibles avec les données de préférences que nous avons pu obtenir du décideur. Par exemple, si le décideur nous a informé que l'alternative a est meilleure que l'alternative b, alors l'ensemble U est restreint aux fonctions u qui vérifient l'inégalité $u(a) \geq u(b)$. Dans l'optique de faire un choix, nous pouvons alors nous intéresser à l'ensemble $\mathrm{ND}_U(\mathcal{X})$ des alternatives non dominées, défini formellement comme suit :

$$\mathrm{ND}_U(\mathcal{X}) = \left\{ x \in \mathcal{X} : \nexists y \in \mathcal{X}, y \succsim_U x \text{ et } \neg(x \succsim_U y) \right\}$$

où $\neg$ est l'opérateur de négation logique et $\succsim_U$ représente la relation binaire définie par : $a \succsim_U b$ si et seulement si $u(a) \geq u(b)$ pour tout $u \in U$. L'ensemble $\mathrm{ND}_U(\mathcal{X})$ constitue en quelque sorte une présélection non "restrictive" pour notre problème. En effet, pour toute alternative $z \notin \mathrm{ND}_U(\mathcal{X})$, il existe une alternative $x \in \mathrm{ND}_U(\mathcal{X})$ vérifiant $u(x) \geq u(z)$ pour tout $u \in U$. Notons toutefois que si l'ensemble $\mathrm{ND}_U(\mathcal{X})$ est trop grand, alors cette présélection ne permet pas de renseigner suffisamment le décideur dans sa problématique de choix. Observons alors que nous avons $\mathrm{ND}_{U'}(\mathcal{X}) \subseteq \mathrm{ND}_U(\mathcal{X})$ pour tout $U' \subseteq U$. Cette observation motive l'approche adoptée par la méthode ISMAUT : le principe général de cette méthode est de poser des questions au décideur, de sorte à réduire progressivement l'ensemble U, jusqu'à ce que le décideur soit en mesure de choisir une alternative dans l'ensemble $\mathrm{ND}_U(\mathcal{X})$. Les auteurs de cette méthode ont par ailleurs montré comment le calcul de $\mathrm{ND}_U(\mathcal{X})$ pouvait être réalisé de manière efficace en utilisant la programmation linéaire (voir [White III et al., 1984] pour plus de détails).

Dans l'optique de limiter le nombre d'interactions avec le décideur, il convient de déterminer, à chaque étape de la procédure d'élicitation, la question permettant de réduire le plus possible la taille de $\mathrm{ND}_U(\mathcal{X})$. Cette problématique, non abordée par la méthode ISMAUT, a motivé par la suite plusieurs extensions de cette dernière ; à titre d'exemple, nous pouvons citer les travaux de Iyengar et al. [2001] et ceux de Ghosh and Kalagnanam [2003] visant à déterminer des questions dont les réponses permettent toujours de "couper en deux" l'espace des paramètres possibles.

Le critère de décision Minimax Regret

Le critère de décision Minimax Regret (e.g., [Savage, 1954, Kouvelis and Yu, 1997]) est généralement utilisé pour la résolution de problèmes de décision dans l'incertain. Plus récemment, ce critère de décision a été employé dans le cadre de problèmes d'optimisation où l'incertitude porte sur les paramètres du modèle décisionnel (e.g., [Salo and Hämäläinen, 2001, Boutilier et al., 2006]). C'est dans ce dernier cadre que nous présentons maintenant ce critère de décision.

Notons $\mathcal{X}$ l'ensemble des alternatives que le décideur doit comparer pour pouvoir prendre sa décision. Supposons que les préférences du décideur sont représentables par une fonction f_ω de paramètres ω, l'inégalité $f_\omega(x) \geq f_\omega(y)$ signifiant que le décideur préfère x à y (aussi noté $x \succsim y$). Par exemple, en décision multicritère, la fonction f_ω pourrait être une somme pondérée (cf. définition 11), les paramètres ω prenant alors la forme d'un vecteur de poids. En décision dans l'incertain, la fonction f_ω pourrait par exemple être une utilité espérée (cf. définition 33), et dans ce cas, les paramètres ω seraient sous la forme d'une fonction d'utilité sur les conséquences. Nous supposons ici que le système de recommandation ne connaît pas l'instance des paramètres permettant de modéliser au mieux les préférences du décideur. À la place, le système prend en considération un ensemble $\mathcal{P}$ contenant des informations sur les préférences du décideur. L'ensemble $\mathcal{P}$ peut par exemple inclure des connaissances a priori sur le problème ou encore des informations obtenues en posant des questions au décideur ; notons que cet ensemble peut être vide si aucune information n'est disponible sur les préférences du décideur. Les données de l'ensemble $\mathcal{P}$ induisent des contraintes sur l'espace des paramètres du modèle. En effet, si par exemple la préférence $x \succsim y$ a été observée, alors la contrainte $f_\omega(x) \geq f_\omega(y)$ doit être imposée pour assurer la compatibilité du modèle avec les préférences du décideur. Notons alors $\Omega_\mathcal{P}$ l'ensemble des instances des paramètres vérifiant les contraintes induites par l'ensemble $\mathcal{P}$; cet ensemble résume en quelque sorte l'incertitude qui règne sur la fonction d'agrégation du décideur. Dans ce contexte d'incertitude, le critère Minimax Regret peut être utilisé pour prendre une décision. Ce critère est caractérisé par les notions suivantes :

Définition 34 (Pairwise Max Regret (PMR))**.** *Le PMR d'une alternative $x \in \mathcal{X}$ par rapport à une alternative $y \in \mathcal{X}$ est défini par :*

$$\mathrm{PMR}(x, y, \Omega_\mathcal{P}) = \max_{\omega \in \Omega_\mathcal{P}} \Big\{ f_\omega(y) - f_\omega(x) \Big\}$$

Par définition, le PMR de $x \in \mathcal{X}$ par rapport à $y \in \mathcal{X}$ est égal à la plus grande perte en terme d'utilité qui puisse être suscitée en recommandant l'alternative x plutôt que l'alternative y. Remarquons que si $\mathrm{PMR}(x, y, \Omega_\mathcal{P}) \leq 0$, alors nous avons forcément $f_\omega(x) \geq f_\omega(y)$ pour tout $\omega \in \Omega_\mathcal{P}$; dans ce cas, nous pouvons conclure que le décideur préfère l'alternative x à l'alternative y, et ce bien que sa fonction d'utilité ne soit pas connue de manière précise.

Définition 35 (Max Regret (MR))**.** *Le MR d'une alternative $x \in \mathcal{X}$ est défini par :*

$$\mathrm{MR}(x, \mathcal{X}, \Omega_\mathcal{P}) = \max_{y \in \mathcal{X}} \mathrm{PMR}(x, y, \Omega_\mathcal{P})$$

Le MR d'une alternative $x \in \mathcal{X}$ est quant à lui égal à la plus grande perte qui puisse être engendrée par la recommandation de x plutôt que toute autre alternative dans $\mathcal{X}$. Notons que la quantité $\mathrm{MR}(x, \mathcal{X}, \Omega_\mathcal{P})$ est nécessairement positive car nous avons en particulier $\mathrm{PMR}(x, x, \Omega_\mathcal{P}) = 0$. Par ailleurs, si $\mathrm{MR}(x, \mathcal{X}, \Omega_\mathcal{P}) = 0$, alors nous savons que l'inégalité $f_\omega(x) \geq f_\omega(y)$ est vraie pour tout $y \in \mathcal{X}$ et tout $\omega \in \Omega_\mathcal{P}$; dans ce cas, nous pouvons conclure que x maximise la fonction d'utilité du décideur. Les solutions optimales au sens du critère de décision Minimax Regret sont par définition celles qui minimisent

la valeur MR. Recommander une telle solution permet en effet de garantir que la perte d'utilité dans le pire cas est minimisée. Plus formellement, cette valeur minimale est définie ci-dessous :

Définition 36 (Minimax Regret (mMR))**.** *Le mMR est défini par :*

$$\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) = \min_{x \in \mathcal{X}} \text{MR}(x, \mathcal{X}, \Omega_{\mathcal{P}})$$

Si une solution optimale au sens du Minimax Regret est recommandée au décideur, alors la valeur mMR fournit une borne supérieure de l'erreur associée à la recommandation (c'est donc une garantie de performance). Soulignons toutefois que ce critère de décision est relativement pessimiste, car celui-ci évalue la qualité d'une recommandation en fonction du pire scénario possible. Des critères de décision dans l'incertain moins conservateurs pourraient être envisagés à la place, comme par exemple le critère d'Hurwicz (e.g., [Hurwicz, 1951, French, 1986]), mais la garantie de performance offerte par le Minimax Regret serait alors perdue. En particulier, lorsque $\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) = 0$, nous savons que toute alternative optimale pour le Minimax Regret maximise nécessairement la fonction d'utilité du décideur.

Élicitation incrémentale fondée sur le critère Minimax Regret

Dans certaines situations, le décideur peut estimer que choisir une alternative optimale au sens du Minimax Regret constitue en soi une prise de décision à risque trop élevé. En effet, lorsque la valeur mMR est très grande, alors la garantie de performance quantifiée par le mMR ne permet pas de borner l'erreur commise dans le pire scénario de manière satisfaisante. Dans ce contexte, le système peut envisager d'acquérir de nouvelles données préférentielles afin de réduire la valeur mMR (et donc améliorer la garantie de performance). Plus précisément, il s'agit de poser des questions au décideur de manière itérative jusqu'à ce que la valeur mMR devienne plus petit qu'un certain seuil $\delta \geq 0$ représentant une borne sur l'erreur que le décideur juge acceptable. C'est uniquement à ce moment là que le système peut recommander au décideur une solution optimale pour le critère Minimax Regret. Une telle procédure est possible car nous avons $\Omega_{\mathcal{P}'} \subseteq \Omega_{\mathcal{P}}$ pour tout ensemble $\mathcal{P}' \supseteq \mathcal{P}$, ce qui implique les inégalités suivantes :

$$\begin{aligned}
&\text{PMR}(x, y, \Omega_{\mathcal{P}'}) \leq \text{PMR}(x, y, \Omega_{\mathcal{P}}), \forall x, y \in \mathcal{X} \\
&\text{MR}(x, \mathcal{X}, \Omega_{\mathcal{P}'}) \leq \text{MR}(x, \mathcal{X}, \Omega_{\mathcal{P}}), \forall x \in \mathcal{X} \\
&\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}'}) \leq \text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})
\end{aligned}$$

De ces inégalités, nous pouvons conclure que la valeur mMR décroît en posant des questions au décideur, et il s'avère que cette valeur décroît même strictement en pratique (e.g., [Braziunas, 2011]). Cette propriété de monotonie fait partie des arguments en faveur de l'utilisation du critère Minimax Regret dans le cadre d'une approche d'élicitation incrémentale ; pour donner un exemple, le critère d'Hurwicz (e.g., [Hurwicz, 1951, French, 1986]) ne vérifie pas cette propriété de monotonie.

L'élicitation incrémentale fondée sur le Minimax Regret se distingue des méthodes à la ISMAUT par sa prise en compte explicite de regrets définis comme des différences d'utilités. Pour aider le décideur

à interpréter les valeurs de regrets, plusieurs options sont envisageables. Nous pouvons par exemple normaliser les valeurs mMR de sorte à associer la valeur 1 au mMR initial (c'est-à-dire, avant toute interaction avec le décideur). En effet, avec cette normalisation, si le mMR est égal à 0.1 à une certaine itération, cela signifie que l'erreur dans le pire scénario a été réduite à 10% de sa valeur initiale. Par ailleurs, lorsque les utilités admettent des équivalents monétaires, le système peut accompagner les valeurs de mMR d'une explication du type "si vous décidez d'arrêter le questionnaire maintenant pour choisir l'alternative recommandée à cette étape, vous risquez de perdre au plus ce montant d'argent".

Pour limiter le nombre d'interactions avec le décideur, il est nécessaire de choisir soigneusement les questions à poser, de manière à faire converger le plus rapidement possible la méthode d'élicitation incrémentale. Différents types de questions peuvent alors être envisagés. Par exemple, les *questions de borne* demandent au décideur si l'utilité globale (ou valeur agrégée) d'une alternative est supérieure ou non à une valeur de référence ; observons toutefois que répondre à une telle question peut être cognitivement difficile pour le décideur car celui-ci doit fournir une évaluation des alternatives en terme d'utilité. Par contraste, les *questions de comparaison* sont relativement simples car celles-ci attendent uniquement du décideur qu'il compare deux alternatives et dise quelle est sa préférée ; ce type de question est par exemple au cœur de la méthode ISMAUT ([White III et al., 1984]). Même en se restreignant à un type de question particulier, il reste de nombreuses questions possibles qu'il faut pouvoir départager. Soulignons toutefois que certaines questions sont vraisemblablement moins informatives que d'autres. Par exemple, dans l'optique de réduire l'incertitude, il est bien évidemment sans intérêt de poser une question dont la réponse est la même pour toutes les instances possibles des paramètres. Par ailleurs, en décision multicritère, demander au décideur de comparer une alternative à une autre qui la domine au sens de Pareto ne nous apprend rien, car les modèles décisionnels considérés sont généralement compatibles avec cette dominance (même remarque pour la dominance stochastique du premier ordre dans le risque). Puisque ces questions ne permettent pas de réduire l'incertitude sur les paramètres du modèle, alors celles-ci ne peuvent induire une réduction du mMR. En revanche, il est probable que la comparaison de deux alternatives de qualité, avec des profils relativement différents, puisse engendrer une réduction importante du mMR. Dans tous les cas, il semble nécessaire de pouvoir évaluer la pertinence d'une question à une itération donnée pour limiter le nombre d'interactions avec le décideur. Idéalement, l'évaluation d'une question à une itération donnée devrait prendre en compte toutes les futures questions et réponses possibles (e.g., [Boutilier, 2002]). Cependant, en pratique, cette évaluation est souvent réalisée de manière "myope", autrement dit en comparant uniquement la valeur mMR avant et après avoir posé la question. Plus précisément, nous pouvons utiliser la notion de valeur de l'information "myope" suivante (e.g., [Viappiani and Boutilier, 2009b]) :

Définition 37 (Worst-case Minimax Regret (WmMR))**.** *Le WmMR d'une question p est défini par :*

$$\mathrm{WmMR}(p, \mathcal{X}, \Omega_{\mathcal{P}}) = \max_{r \in \mathcal{R}_p} \mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P} \cup \{r\}})$$

où $\mathcal{R}_p$ est l'ensemble des réponses possibles à la question p.

À une itération donnée, choisir une question qui minimise le WmMR permet de garantir la plus grande réduction du mMR dans le pire scénario de réponse du décideur. Ce critère de sélection est relativement pessimiste en soi car celui-ci cherche à se prémunir contre la réponse la plus défavorable ; signalons que d'autres critères ont été étudiés dans la littérature, comme par exemple le *maximum average improvement* qui consiste à faire une moyenne des mMR obtenus dans les différents scénarios de réponse (e.g., [Wang and Boutilier, 2003]). Néanmoins, il n'en demeure pas moins que l'évaluation de toutes les questions possibles a généralement un coût non négligeable, notamment lorsque le nombre de questions à envisager est très important. De ce fait, des heuristiques ont été proposées pour la génération de questions qui, en pratique, permettent de réduire efficacement les regrets. Nous présentons ci-après l'heuristique de génération de questions qui est actuellement la plus utilisée pour l'élicitation incrémentale fondée sur le concept de regret.

La stratégie CSS

La stratégie CSS (pour *Current Solution Strategy* en anglais) est une heuristique du critère WmMR qui s'est révélée très efficace en pratique pour la réduction des valeurs mMR (e.g., [Boutilier et al., 2006, Lu and Boutilier, 2011b, Drummond and Boutilier, 2013]). Celle-ci consiste à demander au décideur, à chaque itération, de comparer une alternative $x^* \in \mathcal{X}$ qui est optimale pour le critère Minimax Regret, avec une alternative $y^* \in \mathcal{X}$ maximisant l'erreur commise dans le pire cas lorsque l'alternative x^* est recommandée au décideur (autrement dit, y^* est un élément de $\arg\max_{y \in \mathcal{X}} \text{PMR}(x^*, y, \Omega_\mathcal{P})$). Pour nous convaincre que cette stratégie constitue souvent une bonne heuristique, plaçons nous à une itération quelconque de la procédure d'élicitation. Notons $\mathcal{P}$ l'ensemble de toutes les informations de préférence que nous avons récoltées jusque là. Si $\text{mMR}(\mathcal{X}, \Omega_\mathcal{P}) = 0$ à cette itération, alors la procédure d'élicitation s'arrête ici car l'alternative x^* est nécessairement optimale (par définition du mMR). De ce fait, nous supposons maintenant que $\text{mMR}(\mathcal{X}, \Omega_\mathcal{P}) > 0$ à cette itération. Un premier avantage pratique de cette stratégie est énoncé dans la proposition suivante (e.g., [Benabbou et al., 2015]) :

Proposition 3. *Demander au décideur de comparer x^* et y^* ne peut pas conduire à une incohérence.*

Démonstration. Montrons que l'ensemble $\Omega_\mathcal{P}$ ne peut devenir vide après insertion dans $\mathcal{P}$ de la donnée (y^*, x^*) ou (x^*, y^*) représentant respectivement les préférences $y^* \succsim x^*$ et $x^* \succsim y^*$.

Commençons par la donnée (y^*, x^*). Comme nous avons $\text{PMR}(x^*, y^*, \Omega_\mathcal{P}) = \text{MR}(x^*, \mathcal{X}, \Omega_\mathcal{P})$ par définition de y^* et $\text{MR}(x^*, \mathcal{X}, \Omega_\mathcal{P}) \geq \text{PMR}(x^*, x^*, \Omega_\mathcal{P}) = 0$ par définition des MR, alors nous avons forcément $\text{PMR}(x^*, y^*, \Omega_\mathcal{P}) \geq 0$. Par définition des PMR, nous en déduisons qu'il existe une instance des paramètres $\omega \in \Omega_\mathcal{P}$ telle que $f_\omega(y^*) \geq f_\omega(x^*)$, c'est-à-dire telle que l'alternative y^* est meilleure que l'alternative x^*. Cette dernière déduction permet de conclure que l'ensemble $\Omega_\mathcal{P}$ des paramètres possibles ne sera pas vide après insertion de la donnée (y^*, x^*) dans l'ensemble $\mathcal{P}$.

Considérons maintenant le cas de la donnée (x^*, y^*). Supposons qu'il n'existe aucune instance des paramètres telle que l'alternative x^* est meilleure que l'alternative y^*. Autrement dit, supposons que l'inégalité $f_\omega(x^*) < f_\omega(y^*)$ est vraie pour tout $\omega \in \Omega_\mathcal{P}$. Dans ce cas, par définition des PMR, nous avons

nécessairement $\text{PMR}(y^*, y, \Omega_{\mathcal{P}}) < \text{PMR}(x^*, y, \Omega_{\mathcal{P}})$ pour toute alternative $y \in \mathcal{X}$. Par définition des MR, nous en déduisons ensuite $\text{MR}(y^*, \mathcal{X}, \Omega_{\mathcal{P}}) < \text{MR}(x^*, \mathcal{X}, \Omega_{\mathcal{P}})$. Cette dernière inégalité contredit le fait que l'alternative x^* est optimale pour le critère Minimax Regret. Cette contradiction nous permet de conclure qu'il existe au moins une instance $\omega \in \Omega_{\mathcal{P}}$ pour laquelle l'inégalité $f_\omega(x^*) \geq f_\omega(y^*)$ est vérifiée ; ainsi, l'ensemble $\Omega_{\mathcal{P}}$ des paramètres possibles ne sera pas vide après insertion de (x^*, y^*) dans $\mathcal{P}$. □

Ainsi, avec la CSS, les réponses du décideur ne peuvent jamais provoquer une incohérence avec les informations recueillies précédemment. Soulignons toutefois que cette propriété constitue aussi un inconvénient de cette stratégie, car toute erreur du décideur est indétectable durant l'élicitation (seul le décideur peut s'en rendre compte). Par ailleurs, cette stratégie présente l'intérêt de produire des questions très souvent informatives. Pour le voir, nous étudions ci-après l'évolution de la valeur mMR après insertion dans $\mathcal{P}$ de la donnée (y^*, x^*) ou (x^*, y^*) représentant respectivement les préférences $y^* \succsim x^*$ et $x^* \succsim y^*$:

Cas (y^*, x^*)**.** En insérant la donnée (y^*, x^*) dans $\mathcal{P}$, les paramètres ω sont contraints de vérifier l'inégalité $f_\omega(x^*) \leq f_\omega(y^*)$ aux itérations suivantes. D'après la définition 34, ceci implique :

$$\forall y \in \mathcal{X},\ \text{PMR}(y^*, y, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}}) \leq \text{PMR}(x^*, y, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}}) \tag{1.6}$$

En utilisant ce résultat, nous obtenons :

$$\begin{aligned}
\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}}) &\leq \text{MR}(y^*, \mathcal{X}, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}}) \text{ par définition 36} \\
&= \max_{y\in\mathcal{X}} \text{PMR}(y^*, y, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}}) \text{ par définition 35} \\
&= \max\Big\{0, \max_{y\in\mathcal{X}\setminus\{y^*\}} \text{PMR}(y^*, y, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}})\Big\} \text{ car } \text{PMR}(y^*, y^*, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}}) = 0 \\
&\leq \max\Big\{0, \max_{y\in\mathcal{X}\setminus\{y^*\}} \text{PMR}(x^*, y, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}})\Big\} \text{ d'après l'équation (1.6)} \\
&\leq \max\Big\{0, \max_{y\in\mathcal{X}\setminus\{y^*\}} \text{PMR}(x^*, y, \Omega_{\mathcal{P}})\Big\} \text{ car } \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}} \subseteq \Omega_{\mathcal{P}} \\
&\leq \max\Big\{0, \text{PMR}(x^*, y^*, \Omega_{\mathcal{P}})\Big\} \text{ car } y^* \in \arg\max_{y\in\mathcal{X}} \text{PMR}(x^*, y, \Omega_{\mathcal{P}}) \\
&= \max\Big\{0, \text{MR}(x^*, \mathcal{X}, \Omega_{\mathcal{P}})\Big\} \text{ par définition de } y^* \\
&= \max\Big\{0, \text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})\Big\} \text{ par définition de } x^* \\
&= \text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) \text{ car } \text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) > 0 \text{ par hypothèse}
\end{aligned}$$

Notons que la dernière inégalité devient stricte si l'alternative y^* est l'unique élément de l'ensemble $\arg\max_{y\in\mathcal{X}} \text{PMR}(x^*, y, \Omega_{\mathcal{P}})$. Ceci explique pourquoi, en pratique, la valeur $\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}\cup\{(y^*,x^*)\}})$ est souvent strictement plus petite que la valeur $\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})$ en suivant la CSS.

Cas (x^*, y^*)**.** Après insertion de la donnée (x^*, y^*) dans $\mathcal{P}$, les paramètres ω sont contraints de satisfaire l'inégalité $f_\omega(x^*) \geq f_\omega(y^*)$ aux itérations suivantes. En utilisant la définition 34, nous obtenons alors :

$$\mathrm{PMR}(x^*, y^*, \Omega_{\mathcal{P}\cup\{(x^*,y^*)\}}) \leq 0 \tag{1.7}$$

Ce résultat nous permet de faire les déductions suivantes :

$$\begin{aligned}
\mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P}\cup\{(x^*,y^*)\}}) &\leq \mathrm{MR}(x^*, \mathcal{X}, \Omega_{\mathcal{P}\cup\{(x^*,y^*)\}}) \text{ par définition 36} \\
&= \max_{y\in\mathcal{X}} \mathrm{PMR}(x^*, y, \Omega_{\mathcal{P}\cup\{(x^*,y^*)\}}) \text{ par définition 35} \\
&\leq \max\left\{0, \max_{y\in\mathcal{X}\setminus\{y^*\}} \mathrm{PMR}(x^*, y, \Omega_{\mathcal{P}\cup\{(x^*,y^*)\}})\right\} \text{ d'après l'équation (1.7)} \\
&\leq \max\left\{0, \max_{y\in\mathcal{X}\setminus\{y^*\}} \mathrm{PMR}(x^*, y, \Omega_{\mathcal{P}})\right\} \text{ car } \Omega_{\mathcal{P}\cup\{(x^*,y^*)\}} \subseteq \Omega_{\mathcal{P}} \\
&\leq \max\left\{0, \mathrm{PMR}(x^*, y^*, \Omega_{\mathcal{P}})\right\} \text{ car } y^* \in \arg\max_{y\in\mathcal{X}} \mathrm{PMR}(x^*, y, \Omega_{\mathcal{P}}) \\
&= \max\left\{0, \mathrm{MR}(x^*, \Omega_{\mathcal{P}})\right\} \text{ par définition de } y^* \\
&= \max\left\{0, \mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})\right\} \text{ par définition de } x^* \\
&= \mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) \text{ car } \mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) > 0 \text{ par hypothèse}
\end{aligned}$$

Notons cette fois-ci que la dernière inégalité est stricte si jamais l'alternative y^* est l'unique élément de $\arg\max_{y\in\mathcal{X}} \mathrm{PMR}(x^*, y^*, \Omega_{\mathcal{P}})$. Par conséquent, nous observons $\mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) > \mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P}\cup\{(x^*,y^*)\}})$ relativement souvent en pratique. Finalement, nous avons vu que la stratégie CSS permet de produire des questions informatives car les réponses engendrent généralement une réduction stricte du mMR.

1.5 Conclusion du chapitre

Dans ce chapitre, nous avons pu constater que la modélisation des préférences est une problématique de recherche très étudiée en informatique, mais aussi en économie ou encore en sciences humaines et sociales. En particulier, de nombreux modèles décisionnels ont été proposés dans la littérature, analysés sur la base de propriétés désirées dans l'absolu ; à ce niveau, il est important de souligner que ce chapitre ne présente qu'une faible partie des modélisations existantes. La diversité des modèles décisionnels s'explique notamment par la difficulté de construction d'une modélisation parfaite, que ce soit pour l'aide à la décision multicritère, multi-agents ou dans le risque. Bien que les problèmes abordés dans ces trois domaines soient relativement proches d'un point de vue formel, nous avons pu voir que les résultats obtenus dans un cadre ne se transposent pas directement dans un autre puisque ce ne sont pas les mêmes propriétés qui sont attendues du processus de décision. Par exemple, on peut vouloir modéliser des synergies entre les critères en décision multicritère, imposer une certaine équité en décision collective, ou encore modéliser une certaine attitude vis-à-vis du risque en décision dans le risque. Par ailleurs, il est relativement fréquent de rencontrer des problèmes de décision à l'intersection de ces do-

maines, comme par exemple lorsque plusieurs agents doivent faire un choix parmi un ensemble de loteries dont les conséquences possibles sont évaluées selon plusieurs critères. Dans ces situations, le choix de la modélisation est encore plus complexe, car celui-ci doit tenir compte de toutes les exigences de ces domaines. Néanmoins, en pratique, le choix de la modélisation à utiliser est souvent dirigé par la complexité du modèle en soi, au lieu de privilégier les capacités descriptives. En effet, afin de limiter les temps de résolution et les coûts de communication, les modèles sophistiqués (comme les intégrales de Choquet) sont très souvent délaissés au profit de modélisations plus simples (comme les sommes pondérées), et ce malgré leurs limites descriptives bien connues. C'est pourquoi un des objectifs de cette thèse consiste à proposer des méthodes efficaces pour la résolution de problèmes de décision avec des modèles complexes.

Choisir un modèle décisionnel pour une situation particulière ne suffit généralement pas à décrire les préférences personnelles du décideur impliqué ; par exemple, pour pouvoir utiliser une somme pondérée, il convient de déterminer le jeu de poids permettant de refléter au mieux les préférences du décideur. Il s'agit alors de collecter des informations sur les préférences du décideur pour pouvoir construire une modélisation proche de la réalité. L'élicitation des préférences pour l'aide à la décision est une problématique de recherche visant à concevoir des méthodes qui, en recueillant des informations sur les préférences du décideur, sont capables de formuler une recommandation personnalisée. Dans la littérature, différentes méthodes ont été proposées pour répondre à cette problématique. Cependant, les approches classiques d'élicitation deviennent généralement difficiles à mettre en œuvre en présence de préférences complexes. En effet, le nombre de questions à poser au décideur pour pouvoir éliciter ses préférences augmente avec la complexité du modèle envisagé. Ainsi, un autre objectif de cette thèse consiste à proposer des méthodes d'élicitation de modèles complexes permettant de prendre une décision avec un nombre limité de questions. Par ailleurs, le nombre d'alternatives du problème étant une autre source de complexité, nous nous attellerons à proposer des méthodes d'élicitation efficaces dans le cadre de problèmes de décision sur domaine combinatoire.

Chapitre 2

Élicitation pour la décision multicritère sur domaine non combinatoire

Résumé

Ce chapitre s'articule autour de nos travaux publiés dans [Benabbou et al., 2014, 2015, 2016a, 2017] qui concernent la problématique d'élicitation des préférences dans le cadre de la décision multicritère sur domaine non combinatoire. Nous adoptons ici l'approche d'élicitation incrémentale fondée sur le critère Minimax Regret. Cette approche a été principalement étudiée pour la problématique de choix avec des modèles décisionnels relativement simples. Dans ce chapitre, nous nous intéressons à étendre cette approche à l'intégrale de Choquet dans le cadre des trois grandes problématiques abordées en aide à la décision : le choix (cf. Section 2.2), le rangement (cf. Section 2.3.1) et le tri (cf. Sections 2.3.2 et 2.3.3). La principale difficulté de cette extension réside dans le fait que celle-ci nécessite de résoudre des problèmes d'optimisation faisant intervenir un nombre de variables et de contraintes qui est exponentiel en le nombre de critères du problème. Dans ce chapitre, nous montrons que ces problèmes d'optimisation peuvent être résolus en temps polynomial, à condition de bien choisir les questions à poser au décideur.

Introduction

Dans ce chapitre, nous considérons une situation de décision multicritère quelconque où les préférences du décideur sont imprécisément connues. Pour aider le décideur à prendre une décision, nous adoptons ici l'approche d'élicitation incrémentale fondée sur le critère Minimax regret (que nous avons présentée en section 1.4.2). Notre objectif est de parvenir à étendre cette approche aux intégrales de Choquet de manière efficace. Durant la présentation de nos travaux allant dans cette direction, nous utilisons les mêmes notations que celles introduites en section 1.1. Plus précisément, nous notons $\mathcal{X}$ l'ensemble des alternatives (e.g., objets, candidats, actions potentielles) que le système doit comparer pour aider le décideur à prendre une décision. Chaque alternative du problème est évaluée sur un ensemble de critères $\mathcal{Q} = \{1, \ldots, q\}$ que le décideur juge important de considérer (e.g., couleur, qualité, temps, précision, revenu). Toute alternative $x \in \mathcal{X}$ est caractérisée par un vecteur de performances $x = (x_1, \ldots, x_q)$ supposé connu, où $x_j \in [0, 1]$ représente l'utilité de l'alternative x sur le critère j, avec $j \in \mathcal{Q}$; ces échelles cardinales peuvent par exemple avoir été obtenues en utilisant au préalable une méthode à la MACBETH dont l'idée clé est d'interroger le décideur non seulement sur des préférences relatives mais aussi sur des intensités de préférences (e.g., [Bana e Costa and Vansnick, 1994, 2000]). Nous supposons que les préférences du décideur sont représentables par une fonction d'agrégation paramétrique dont les paramètres sont imprécisément connus ; sans perte de généralité, nous supposons par ailleurs que cette fonction est à maximiser.

Rappelons que l'élicitation incrémentale fondée sur le Minimax Regret requiert, à chaque itération, un calcul de mMR sur l'ensemble $\mathcal{X}$ des alternatives possibles. Par définition des mMR (cf. définition 36), ce calcul peut être réalisé par le biais d'au plus $|\mathcal{X}| \times (|\mathcal{X}| - 1)$ calculs de PMR. Ce nombre peut toutefois être réduit de manière significative en pratique, en utilisant un élagage de type alpha-beta qui est une technique classique pour améliorer les performances de l'algorithme minimax en théorie des jeux. Empiriquement, le nombre de calculs de PMR devient presque linéaire avec cet élagage, mais celui-ci reste bien évidemment quadratique dans le pire cas (e.g., [Braziunas, 2011, Viappiani and Boutilier, 2009a]). Par conséquent, il est d'une importance majeure de bénéficier de méthodes permettant de résoudre efficacement les problèmes PMR. Remarquons que ces derniers problèmes d'optimisation ont une formulation différente selon le modèle décisionnel considéré ; en particulier, ces problèmes se complexifient avec la sophistication du modèle. Ainsi, une première difficulté liée à l'extension de cette approche à des modèles décisionnels complexes provient du calcul des PMR à chaque itération de la procédure.

2.1 Quelques résultats introductifs

Dans cette section, nous présentons des méthodes permettant de calculer efficacement la valeur $\text{PMR}(x, y, \Omega_{\mathcal{P}})$, avec $x, y \in \mathcal{X}$, pour des fonctions d'agrégation simples, où $\mathcal{P}$ représente l'ensemble des informations disponibles sur les préférences du décideur et $\Omega_{\mathcal{P}}$ est l'ensemble des instances des paramètres compatibles avec ces informations. Pour simplifier la présentation, nous supposons que l'ensemble $\mathcal{P}$ est uniquement constitué de paires (a, b) représentant la donnée "je préfère l'alternative a à l'alternative b".

La somme pondérée

Lorsque les préférences du décideur sont modélisées par une somme pondérée (cf. définition 11), l'ensemble $\Omega_{\mathcal{P}}$ est composé de tous les jeux de poids normalisés $\omega = (\omega_1, \ldots, \omega_q)$ vérifiant la contrainte $\sum_{j\in\mathcal{Q}} \omega_j a_j \geq \sum_{j\in\mathcal{Q}} \omega_j b_j$ pour tout $(a, b) \in \mathcal{P}$. Avec une somme pondérée, le problème d'optimisation que nous souhaitons résoudre efficacement est le suivant :

$$\mathrm{PMR}(x, y, \Omega_{\mathcal{P}}) = \max_{\omega\in\Omega_{\mathcal{P}}} \Big\{ \mathrm{SP}(y, \omega) - \mathrm{SP}(x, \omega) \Big\} = \max_{\omega\in\Omega_{\mathcal{P}}} \sum_{j\in\mathcal{Q}} \Big(\omega_j y_j - \omega_j x_j \Big)$$

Puisque la fonction à maximiser est linéaire en les paramètres du modèle et que l'ensemble $\Omega_{\mathcal{P}}$ est décrit par des contraintes linéaires, alors calculer $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}})$ pour une somme pondérée revient à résoudre le programme linéaire suivant :

$$\begin{aligned}
\max_{\omega} \quad & \sum_{j\in\mathcal{Q}} \Big(\omega_j y_j - \omega_j x_j \Big) \\
s.c. \quad & \sum_{j\in\mathcal{Q}} \omega_j = 1 && (2.1)\\
& \sum_{j\in\mathcal{Q}} \omega_j a_j \geq \sum_{j\in\mathcal{Q}} \omega_j b_j, \ \forall (a, b) \in \mathcal{P} && (2.2)\\
& \omega_j \geq 0, \ \forall j \in \mathcal{Q} && (2.3)
\end{aligned}$$

où les équations (2.1) et (2.3) servent à contraindre le vecteur de poids ω à être positif et normalisé, tandis que les contraintes décrites par l'équation (2.2) assurent la compatibilité du modèle avec les préférences observées. Comme ce programme linéaire comprend uniquement q variables (une variable ω_j par critère $j \in \mathcal{Q}$) et un faible nombre de contraintes (de l'ordre de $|\mathcal{P}|$), celui-ci peut être résolu en temps polynomial (par rapport au nombre de critères) à l'aide d'un solveur de programmation linéaire.

La moyenne ordonnée pondérée (OWA)

Le modèle OWA est paramétré par un vecteur de poids normalisé $\omega = (\omega_1, \ldots, \omega_q)$ associant une pondération aux rangs plutôt qu'aux critères (contrairement à la somme pondérée). Plus précisément, sous l'hypothèse du modèle OWA, l'utilité d'une alternative $a \in \mathcal{X}$ est définie par :

$$\mathrm{OWA}(a, \omega) = \sum_{j\in\mathcal{Q}} \omega_j a_{(j)}$$

où $(.)$ est une permutation des critères vérifiant $a_{(1)} \leq \ldots \leq a_{(q)}$ (cf. définition 12). Ainsi, $\Omega_{\mathcal{P}}$ est l'ensemble des jeux de poids normalisés $\omega = (\omega_1, \ldots, \omega_q)$ tels que la contrainte $\sum_{j\in\mathcal{Q}} \omega_j a_{(j)} \geq \sum_{j\in\mathcal{Q}} \omega_j b_{(j)}$ est vérifiée pour tout $(a, b) \in \mathcal{P}$; notons que ces contraintes sont linéaires en les poids $\omega_j, j \in \mathcal{Q}$. Pour ce modèle décisionnel, il s'agit pour nous de déterminer efficacement la valeur suivante :

$$\mathrm{PMR}(x, y, \Omega_{\mathcal{P}}) = \max_{\omega \in \Omega_{\mathcal{P}}} \Big\{ \mathrm{OWA}(y, \omega) - \mathrm{OWA}(x, \omega) \Big\} = \max_{\omega \in \Omega_{\mathcal{P}}} \sum_{j \in \mathcal{Q}} \Big(\omega_j y_{(j)} - \omega_j x_{(j)} \Big)$$

Remarquons que la fonction à maximiser est très similaire à celle qu'il faut optimiser sous l'hypothèse d'une somme pondérée. De ce fait, en utilisant les mêmes arguments que pour ce dernier modèle, nous pouvons calculer la valeur $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}})$ au sens de OWA en résolvant le programme linéaire suivant :

$$\begin{aligned}
\max_{\omega} \quad & \sum_{j \in \mathcal{Q}} \Big(\omega_j y_{(j)} - \omega_j x_{(j)} \Big) \\
s.c. \quad & \sum_{j \in \mathcal{Q}} \omega_j = 1 && (2.4) \\
& \omega_j - \omega_{j+1} \geq 0, \ \forall j \in \{1, \ldots, q-1\} && (2.5) \\
& \sum_{j \in \mathcal{Q}} \omega_j a_{(j)} \geq \sum_{j \in \mathcal{Q}} \omega_j b_{(j)}, \ \forall (a, b) \in \mathcal{P} && (2.6) \\
& \omega_j \geq 0, \ \forall j \in \mathcal{Q} && (2.7)
\end{aligned}$$

où les équations (2.4) et (2.7) permettent de garantir la normalisation des poids et l'équation (2.6) assure la compatibilité du vecteur de poids avec les préférences déjà observées. La contrainte décrite par l'équation (2.5) de ce programme permet, quant à elle, de contraindre les poids à être décroissants, afin de favoriser l'équité entre les critères. Cette propriété est particulièrement désirable lorsque les différents critères représentent les points de vue de plusieurs agents ; il convient toutefois de supprimer cette équation du programme linéaire lorsque cette propriété n'est pas souhaitée. Observons que cette formulation linéaire fait intervenir uniquement q variables (une variable ω_j par rang) et un faible nombre de contraintes (de l'ordre de $|\mathcal{P}|$). Par conséquent, calculer la valeur $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}})$ au sens de OWA peut être effectué en temps polynomial (par rapport au nombre de critères) en utilisant un solveur de programmes linéaires.

L'intégrale de Choquet

Pour les intégrales de Choquet, les paramètres sont sous la forme d'une capacité (cf. définition 13) qui, rappelons-le, est une fonction d'ensemble $v : 2^{\mathcal{Q}} \to [0, 1]$ normalisée (c'est-à-dire, $v(\emptyset) = 0$ et $v(\mathcal{Q}) = 1$) et monotone (c'est-à-dire, $v(A) \leq v(A \cup \{j\})$ pour tous $A \subset \mathcal{Q}, j \in \mathcal{Q} \backslash A$). Pour ce modèle décisionnel, l'utilité d'une alternative $a \in \mathcal{X}$ est définie par :

$$\mathrm{Ch}(a, v) = \sum_{j \in \mathcal{Q}} \big[a_{(j)} - a_{(j-1)} \big] \, v(A_{(j)})$$

où $(.)$ est une permutation des critères telle que $a_{(1)} \leq \ldots \leq a_{(q)}$, $a_{(0)} = 0$ et $A_{(j)} = \{(j), \ldots, (q)\}$ (cf. définition 16). Il est important de remarquer que, bien que $\mathrm{Ch}(a, v)$ ne soit pas une fonction linéaire en a avec v fixée (par exemple, $\mathrm{Ch}(a + a', v) \neq \mathrm{Ch}(a, v) + \mathrm{Ch}(a', v)$ en général), la fonction $\mathrm{Ch}(a, v)$ est linéaire

en v pour a fixée car la permutation (.) est alors fixée. De ce fait, l'ensemble $\Omega_{\mathcal{P}}$ est ici composé de toutes les capacités v vérifiant la contrainte linéaire $\sum_{j\in\mathcal{Q}}\left[a_{(j)}-a_{(j-1)}\right]v(A_{(j)}) \geq \sum_{j\in\mathcal{Q}}\left[b_{(j)}-b_{(j-1)}\right]v(B_{(j)})$ pour toute paire $(a,b)\in\mathcal{P}$. Notre objectif est de calculer efficacement la valeur suivante :

$$\begin{aligned}\mathrm{PMR}(x,y,\Omega_{\mathcal{P}}) &= \max_{v\in\Omega_{\mathcal{P}}}\Big\{\mathrm{Ch}(y,v)-\mathrm{Ch}(x,v)\Big\}\\ &= \max_{v\in\Omega_{\mathcal{P}}}\sum_{j\in\mathcal{Q}}\Big(\left[y_{(j)}-y_{(j-1)}\right]v(Y_{(j)})-\left[x_{(j)}-x_{(j-1)}\right]v(X_{(j)})\Big)\end{aligned}$$

Comme une intégrale de Choquet est une fonction linéaire en v, le calcul de $\mathrm{PMR}(x,y,\Omega_{\mathcal{P}})$ peut être réalisé en utilisant la programmation linéaire. Plus précisément, en notant v_A la variable représentant la valeur $v(A)$ pour tout $A\subseteq\mathcal{Q}$, il s'agit de résoudre le programme linéaire suivant :

$$\max_{v} \quad \sum_{j\in\mathcal{Q}}\Big(\left[y_{(j)}-y_{(j-1)}\right]v_{Y_{(j)}}-\left[x_{(j)}-x_{(j-1)}\right]v_{X_{(j)}}\Big) \tag{2.8}$$

$$s.c. \quad v_{\emptyset}=0,\ v_{\mathcal{Q}}=1 \tag{2.9}$$

$$v_A \leq v_{A\cup\{j\}},\ \forall A\subset\mathcal{Q},\forall j\in\mathcal{Q}\backslash A \tag{2.10}$$

$$\sum_{j\in\mathcal{Q}}\left[a_{(j)}-a_{(j-1)}\right]v_{A_{(j)}} \geq \sum_{j\in\mathcal{Q}}\left[b_{(j)}-b_{(j-1)}\right]v_{B_{(j)}},\ \forall(a,b)\in\mathcal{P} \tag{2.11}$$

où les équations (2.9) et (2.10) permettent de garantir respectivement que v est une fonction normalisée et monotone (c'est-à-dire, une capacité), alors que l'équation (2.11) oblige cette fonction à être compatible avec les données stockées dans $\mathcal{P}$. Bien que ce programme soit linéaire en ses variables (v_A, $A\subseteq\mathcal{Q}$), ces dernières sont en nombre exponentiel par rapport au nombre de critères (exactement 2^q), ce qui est aussi le cas du nombre de contraintes de monotonie (ici égal à $\sum_{k=0}^{q-1}(q-k)\times\binom{q}{k}$). En conséquence, contrairement aux programmes linéaires vus précédemment, utiliser un solveur pour résoudre ce programme est envisageable uniquement pour des problèmes de décision impliquant un faible nombre de critères.

Pour contourner ce problème, une approche alternative a récemment été proposée pour l'intégrale de Choquet, consistant à calculer directement le MR d'une solution $x\in\mathcal{X}$ sans passer par les calculs des PMR (e.g., [Timonin, 2013]). Plus précisément, il s'avère que le calcul de MR revient à maximiser une fonction convexe sur un ensemble convexe :

$$\begin{aligned}\mathrm{MR}(x,\mathcal{X},\Omega_{\mathcal{P}}) &= \max_{y\in\mathcal{X}}\mathrm{PMR}(x,y,\Omega_{\mathcal{P}}) \text{ par définition 35}\\ &= \max_{y\in\mathcal{X}}\max_{v\in\Omega_{\mathcal{P}}}\Big\{\mathrm{Ch}(y,v)-\mathrm{Ch}(x,v)\Big\} \text{ par définition 34}\\ &= \max_{v\in\Omega_{\mathcal{P}}}\max_{y\in\mathcal{X}}\Big\{\mathrm{Ch}(y,v)-\mathrm{Ch}(x,v)\Big\}\end{aligned}$$

En effet, $\max_{y\in\mathcal{X}}\{\mathrm{Ch}(y,v)-\mathrm{Ch}(x,v)\}$ est une fonction convexe en v et $\Omega_{\mathcal{P}}$ est un ensemble convexe. En utilisant le fait bien connu que la valeur maximale d'une telle fonction est atteinte en un point extrême

de l'ensemble, nous obtenons le résultat suivant :

$$\mathrm{MR}(x, \mathcal{X}, \Omega_{\mathcal{P}}) = \max_{v \in \Omega^*_{\mathcal{P}}} \max_{y \in \mathcal{X}} \Big\{ \mathrm{Ch}(y, v) - \mathrm{Ch}(x, v) \Big\}$$

où $\Omega^*_{\mathcal{P}}$ représente l'ensemble des points extrêmes de $\Omega_{\mathcal{P}}$. Ce résultat est très intéressant car permet en théorie de réduire le problème d'optimisation en une série de différences de valeurs (une différence par point extrême). Cependant, cette approche est en pratique très coûteuse, car le nombre de points extrêmes est généralement trop important pour permettre de les énumérer de manière exhaustive (e.g., [Combarro and Miranda, 2008, Grabisch and Miranda, 2008]).

Dans le premier chapitre, nous avons vu que ce modèle décisionnel admet une formulation alternative fondée sur l'inverse de Möbius $m : 2^{\mathcal{Q}} \to \mathbb{R}$ associée à la capacité v (cf. définition 17). Plus précisément, l'utilité d'une alternative $a \in \mathcal{X}$ peut se réécrire de la manière suivante (cf. définition 18) :

$$\mathrm{Ch}(a, v) = \sum_{A \subseteq \mathcal{Q}} m(A) \min_{j \in A} \{a_j\}$$

Ainsi, en notant m_A la variable représentant la masse $m(A)$ pour tout $A \subseteq \mathcal{Q}$, la quantité $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}})$ peut être déterminée en résolvant le programme linéaire suivant :

$$\max_{m} \quad \sum_{A \subseteq \mathcal{Q}} \Big(m_A \min_{j \in A} \{y_j\} - m_A \min_{j \in A} \{x_j\} \Big) \tag{2.12}$$

$$s.c. \quad m_\emptyset = 0, \ \sum_{A \subseteq \mathcal{Q}} m_A = 1 \tag{2.13}$$

$$\sum_{B : \{j\} \subseteq B \subseteq A} m_B \geq 0, \ \forall A \subseteq \mathcal{Q}, \forall j \in A \tag{2.14}$$

$$\sum_{A \subseteq \mathcal{Q}} m_A \min_{j \in A} \{a_j\} \geq \sum_{A \subseteq \mathcal{Q}} m_A \min_{j \in A} \{b_j\}, \ \forall (a, b) \in \mathcal{P} \tag{2.15}$$

où l'équation (2.13) permet de garantir que v est une fonction normalisée, l'équation (2.14) assure la monotonie (comme déjà observé dans d'autres travaux, voir par exemple [Chateauneuf and Jaffray, 1989, Meyer and Roubens, 2006]), et l'équation (2.15) oblige cette fonction à être compatible avec les données collectées. Sans surprise, le nombre de variables et de contraintes de monotonie augmente ici aussi de façon exponentielle par rapport au nombre de critères. Néanmoins, nous allons voir maintenant que, pour des sous-classes de capacité (fonction de croyance, capacité 2-additive), ce dernier programme linéaire peut être simplifié.

Fonctions de croyance. Les fonctions de croyance sont par définition les capacités de Choquet monotones d'ordre infini (e.g., [Dempster, 1967, Shafer, 1976]). Comme les masses de Möbius associées à une fonction de croyance sont toutes positives, alors les contraintes de monotonie, exprimées par l'équation (2.14), sont naturellement vérifiées par ces fonctions. De ce fait, pour les fonctions de croyance, cette

dernière équation peut être supprimée du programme linéaire en imposant à la place que les variables $m_A, A \subseteq \mathcal{Q}$, soient toutes positives. Ceci conduit à une formulation linéaire avec un nombre de contraintes de l'ordre de $|\mathcal{P}|$. Cependant, le nombre de variables impliquées dans cette formulation reste exponentiel en le nombre de critères du problème.

Capacités 2-additives. Une capacité 2-additive vérifie $m(A) = 0$ pour tout sous-ensemble $A \subseteq \mathcal{Q}$ contenant au moins trois éléments (cf. définition 19). Par conséquent, sous cette hypothèse, le programme linéaire (2.12)-(2.15) peut être simplifié comme suit :

$$\begin{aligned}
\max_{m} \quad & \sum_{A\subseteq\mathcal{Q}:|A|\leq 2} \Big(m_A \min_{j\in A}\{y_j\} - m_A \min_{j\in A}\{x_j\} \Big) \\
s.c. \quad & m_\emptyset = 0, \quad \sum_{j\in\mathcal{Q}} m_{\{j\}} + \sum_{j,k\in\mathcal{Q}:j<k} m_{\{j,k\}} = 1 && (2.16) \\
& \sum_{k\in A} m_{\{j,k\}} \geq 0, \ \forall j \in \mathcal{Q}, \ \forall A \subseteq \mathcal{Q} : \{j\} \subseteq A && (2.17) \\
& \sum_{A\subseteq\mathcal{Q}:|A|\leq 2} m_A \min_{j\in A}\{a_j\} \geq \sum_{A\subseteq\mathcal{Q}:|A|\leq 2} m_A \min_{j\in A}\{b_j\}, \ \forall (a,b) \in \mathcal{P}
\end{aligned}$$

Bien que le nombre de variables impliquées ne soit pas exponentiel (exactement égal à $(q^2+q)/2$ ici), ce n'est pas le cas du nombre de contraintes de monotonie exprimées par l'équation (2.17). Afin de réduire le nombre de contraintes de monotonie, nous pouvons par exemple utiliser le fait que l'ensemble des capacités normalisées forment un polyèdre convexe dans le cas 2-additif et que chaque capacité s'exprime comme une combinaison convexe de ses q^2 points extrêmes ; ce résultat a récemment été exploité pour la régression logistique fondée sur l'intégrale de Choquet ou *Choquistic regression* en anglais (e.g., [Hüllermeier and Fallah Tehrani, 2013]). Pour définir formellement les inverses de Möbius associées à ces q^2 capacités, nous utilisons les ensembles $\mathcal{Q}_2 = \{X \subseteq \mathcal{Q} : |X| = 2\}$ et $\mathcal{Q}_{1,2} = \{X \subseteq \mathcal{Q} : 1 \leq |X| \leq 2\}$. Plus précisément, chaque élément X de l'ensemble $\mathcal{Q}_2$ permet de définir l'inverse de Möbius m_2^X associée à un point extrême de la manière suivante :

$$\forall A \subseteq \mathcal{Q}, \ m_2^X(A) = \begin{cases} 1 & \text{si } A = X \\ 0 & \text{sinon} \end{cases}$$

Les éléments X de l'ensemble $\mathcal{Q}_{1,2}$ caractérisent les inverses de Möbius $m_{1,2}^X$ des autres points extrêmes :

$$\forall A \subseteq \mathcal{Q}, \ m_{1,2}^X(A) = \begin{cases} 1 & \text{if } A \neq \emptyset \text{ et } A \subset X \\ -1 & \text{si } A = X \\ 0 & \text{sinon} \end{cases}$$

En utilisant ces q^2 fonctions, l'inverse de Möbius m associée à une capacité 2-additive quelconque peut toujours s'écrire sous la forme suivante : $m = \sum_{X\in\mathcal{Q}_2} \alpha_2^X m_2^X + \sum_{X\in\mathcal{Q}_{1,2}} \alpha_{1,2}^X m_{1,2}^X$, où α_2^X, $X \in \mathcal{Q}_2$, $\alpha_{1,2}^X$, $X \in \mathcal{Q}_{1,2}$, sont des réels positifs sommant à 1. Ce résultat nous autorise finalement à remplacer les

contraintes de monotonie (2.17) et de normalisation (2.16) par les $(3q^2+q+2)/2$ contraintes suivantes :

$$m_A = \sum_{X \in \mathcal{Q}_2} \alpha_2^X m_2^X(A) + \sum_{X \in \mathcal{Q}_{1,2}} \alpha_{1,2}^X m_{1,2}^X(A),\ \forall A \subseteq \mathcal{Q},\ |A| \leq 2$$

$$\alpha_2^X \geq 0,\ \forall X \in \mathcal{Q}_2$$

$$\alpha_{1,2}^X \geq 0,\ \forall X \in \mathcal{Q}_{1,2}$$

$$\sum_{X \in \mathcal{Q}_2} \alpha_1^X + \sum_{X \in \mathcal{Q}_{1,2}} \alpha_2^X = 1$$

Finalement, au prix de l'ajout de q^2 variables correspondant aux poids de la combinaison convexe (c'est-à-dire, α_2^X, $X \in \mathcal{Q}_2$, $\alpha_{1,2}^X$, $X \in \mathcal{Q}_{1,2}$), nous obtenons un programme linéaire plus compact, comprenant seulement un nombre quadratique de contraintes et de variables.

Quelques résultats expérimentaux

Les expériences suivantes ont pour objectif de quantifier le temps nécessaire au calcul du mMR à chaque itération de la procédure d'élicitation en fonction du modèle décisionnel considéré. Les temps de calcul sont donnés en secondes, pour des expériences réalisées sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire. Dans ces expériences, nous utilisons le solveur Gurobi pour résoudre les programmes linéaires introduits dans cette section pour le calcul des PMR. Nous reportons les résultats obtenus avec les modèles décisionnels suivants :

- SP : somme pondérée, sans aucune restriction sur les jeux de poids,
- OWA : OWA avec des poids décroissants (modélisant des préférences pour les solutions équilibrées),
- Ch-2add : intégrale de Choquet définie à partir d'une fonction de croyance 2-additive.

Pour évaluer l'impact du nombre de données stockées dans $\mathcal{P}$, nous reportons les temps de calcul obtenus à différentes itérations du processus d'élicitation incrémentale ; pour la génération de questions, nous avons utilisé la stratégie CSS (décrite en section 1.4.2) et les réponses aux questions ont été simulées en utilisant une instance aléatoire des paramètres du modèle considéré. Par ailleurs, nous faisons aussi varier q le nombre de critères et $|\mathcal{X}|$ le nombre d'alternatives du problème. Pour engendrer des instances difficiles, nous avons considéré le problème de sac à dos multi-objectifs (qui est un problème standard en optimisation combinatoire) :

$$\begin{aligned} \max_x \quad & (\sum_{i=1}^n x_i v_i^1, \ldots, \sum_{i=1}^n x_i v_i^q) \\ s.c. \quad & \sum_{i=1}^n w_i x_i \leq W \\ & x_i \in \{0,1\},\ \forall i \in \{1,\ldots,n\} \end{aligned}$$

où l'opérateur max représente la dominance de Pareto, n est le nombre d'objets, w_i est le poids de $i^{\text{ème}}$

objet, W est la capacité du sac à dos, x_i est la variable de décision associée à l'objet i et v_i^j est l'utilité de l'objet i sur le critère j. Plus précisément, les ensembles $\mathcal{X}$ que nous considérons sont composés de solutions Pareto-optimales d'instances aléatoires du problème de sac à dos multi-objectifs. Les résultats donnés ci-dessous ont tous été obtenus en moyennant les temps observés sur 30 tests.

TABLE 2.1 – Comparaison des temps de calcul (en secondes) observés pour la détermination du mMR.

q	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{P}\rvert$	SP	OWA	Ch-2add
5	1000	0	1.19	1.34	1.21
5	1000	20	2.97	2.41	4.74
5	5000	0	6.02	7.05	6.35
5	5000	20	17.13	12.29	32.24
10	1000	0	1.21	1.40	1.46
10	1000	20	3.09	2.83	4.67
10	5000	0	6.72	7.75	9.24
10	5000	20	27.12	17.14	49.01

Dans la table 2.1, nous voyons tout d'abord que les temps de calcul obtenus avec le modèle Ch-2add sont globalement plus importants que tous les autres. Ce fait n'est pas vraiment surprenant puisque ce modèle possède plus de paramètres que les autres modèles considérés (et donc plus de variables dans les programmes linéaires calculant les PMR) ; plus précisément, le nombre de paramètres de Ch-2add est en $O(q^2)$ contre $O(q)$ pour les autres modèles. Par ailleurs, pour tous les modèles considérés, les temps de résolution augmentent globalement avec le nombre de données contenues dans $\mathcal{P}$. Ce résultat n'est pas particulièrement étonnant puisque chaque donnée ajoute une contrainte supplémentaire dans les programmes linéaires. Le plus intéressant, c'est de remarquer que les temps de calcul sont relativement raisonnables avec 1000 alternatives (le décideur n'attend pas plus que 5 secondes entre chaque question), mais ne le sont plus vraiment avec 5 fois plus d'alternatives. Ceci illustre le fait que cette approche, fondée sur le calcul systématique des PMR, n'est pas directement applicable sur les problèmes d'optimisation combinatoire (ici le sac à dos multi-objectifs). Dans le chapitre suivant, nous verrons comment cette approche peut être étendue de manière efficace dans le cadre de problèmes de décision multicritères sur domaine combinatoire.

Dans cette section, nous avons étudié la problématique de calcul des PMR pour des sous-classes de capacité. Il reste néanmoins difficile de savoir a priori si les préférences du décideur peuvent être modélisées à l'aide de ces capacités très spécifiques. C'est pourquoi nous nous intéressons maintenant au calcul des PMR dans le cas général, autrement dit sans faire aucune hypothèse sur la forme des capacités envisageables (si ce n'est qu'elles sont normalisées et monotones par rapport à l'inclusion).

2.2 Approche incrémentale pour le choix avec l'intégrale de Choquet

La plupart des travaux sur l'élicitation d'une intégrale de Choquet s'inscrivent dans le cadre de l'approche que nous avons appelée "apprentissage à partir d'une base données" (cf. section 1.4.1). Plus précisément, il s'agit de considérer en entrée une base de données statique contenant des informations sur les préférences du décideur, et de se concentrer sur la détermination des valeurs de capacité permettant de représenter au mieux cette base de données. Par exemple, on peut avoir envie de déterminer les valeurs de capacité qui permettent de minimiser une erreur quadratique entre des valeurs de Choquet et des valeurs d'utilité prescrites par le décideur sur un échantillon d'alternatives de référence. On peut aussi imposer des contraintes sur l'intégrale de Choquet pour que celle-ci soit compatible avec un ordre partiel sur un sous-ensemble d'alternatives. Cette approche pour l'élicitation d'une intégrale de Choquet est illustrée dans de nombreux articles (e.g., [Grabisch et al., 1995, Marichal and Roubens, 2000, Meyer and Roubens, 2006, Grabisch et al., 2009, Tehrani et al., 2012, Hüllermeier and Fallah Tehrani, 2013]) et intégrée dans des systèmes d'aide à la décision, comme par exemple TOMASO [Marichal et al., 2005] et MYRIAD [Huédé et al., 2006].

Dans cette section, nous nous intéressons à étendre efficacement l'approche incrémentale fondée sur le Minimax Regret aux intégrales de Choquet. Plus précisément, nous allons montrer comment le programme linéaire (2.8)-(2.11), correspondant au calcul de $\text{PMR}(x, y, \Omega_{\mathcal{P}})$ avec l'intégrale de Choquet, peut être simplifié et s'exprimer avec seulement un nombre linéaire de variables et de contraintes de monotonie, à condition de restreindre la forme des questions possibles durant l'élicitation (autrement dit, lorsque les données stockées dans $\mathcal{P}$ ont une forme particulière). Puis, nous proposerons un algorithme en $O(q^2)$ permettant de résoudre ce programme simplifié sans faire appel à un solveur de programmes linéaires. Afin de pouvoir mettre en œuvre ces résultats théoriques plutôt intéressants, nous proposerons ensuite une stratégie de génération de questions permettant de collecter des données de la forme souhaitée, tout en réduisant de manière efficace la valeur mMR en pratique.

2.2.1 Optimisation du critère Minimax Regret par programmation linéaire

Dans cette section, nous notons $\mathcal{A}_{(x,y)}$ l'ensemble constitué de tous les ensembles de niveau associés aux alternatives x et y (cf. définition 15). Plus formellement, l'ensemble $\mathcal{A}_{(x,y)}$ est défini par :

$$\mathcal{A}_{(x,y)} = \{X_{(j)} : j \in \mathcal{Q}\} \cup \{Y_{(j)} : j \in \mathcal{Q}\}$$

Observons que l'ensemble $\mathcal{A}_{(x,y)}$ contient au plus $2q - 1$ éléments et que ces derniers correspondent aux sous-ensembles de critères apparaissant dans la fonction objectif du programme (2.8)-(2.11) ; la fonction objectif de ce programme linéaire s'exprime donc toujours avec au plus $2q - 1$ variables. Cette simple observation constitue en réalité le point de départ de notre réflexion sur la simplification de ce programme : nous allons montrer que, sous certaines hypothèses sur la forme des données stockées dans $\mathcal{P}$, toutes les autres variables peuvent être retirées du programme linéaire. Plus précisément, nous supposons à présent que les données contenues dans l'ensemble $\mathcal{P}$ ont toutes été obtenues en demandant

au décideur de bien vouloir comparer :

- une alternative binaire de type $1A0$, $A \subseteq \mathcal{Q}$, représentant une solution fictive avec une utilité maximale sur tous les critères dans A et une utilité minimale sur tous les critères dans $\mathcal{Q}\backslash A$,
- à un profil d'utilité constant de type $\Lambda = (\lambda, \ldots, \lambda) \in [0,1]^q$, représentant une solution fictive d'utilité constante égale à λ sur tous les critères.

Par exemple, pour un problème de décision avec 4 critères (c'est-à-dire, $\mathcal{Q} = \{1,2,3,4\}$), nous avons $1A0 = (1,0,0,1)$ quand $A = \{1,4\}$ et $1A0 = (0,1,1,1)$ lorsque $A = \{2,3,4\}$. Ces alternatives sont particulièrement intéressantes pour notre problème d'optimisation car, pour toute capacité v, nous avons :

$$\mathrm{Ch}(1A0, v) = v(A)$$

pour tout $A \subseteq \mathcal{Q}$, ce résultat découlant directement de la définition 16. À titre illustratif, pour $\mathcal{Q} = \{1,2,3,4\}$ et $A = \{1,4\}$, nous avons : $\mathrm{Ch}(1A0, v) = 0 \times v(\mathcal{Q}) + (0-0) \times v(\{1,3,4\}) + (1-0) \times v(\{1,4\}) + (1-1) \times v(\{4\}) = v(\{1,4\}) = v(A)$, en prenant la permutation des critères $(.)$ définie par $(1) = 2$, $(2) = 3$, $(3) = 1$ et $(4) = 4$. Les alternatives de type $\Lambda = (\lambda, \ldots, \lambda) \in [0,1]^q$ ont, quant à elles, la particularité de vérifier l'égalité suivante :

$$\mathrm{Ch}(\Lambda, v) = \lambda$$

Ce résultat est aussi une conséquence directe de la définition 16. Par exemple, pour $\mathcal{Q} = \{1,2,3,4\}$ et $\lambda = 0.5$, nous obtenons : $\mathrm{Ch}((0.5, 0.5, 0.5, 0.5), v) = 0.5 \times v(\mathcal{Q}) + (0.5 - 0.5) \times v(\{2,3,4\}) + (0.5 - 0.5) \times v(\{3,4\}) + (0.5 - 0.5) \times v(\{4\}) = 0.5 \times v(\mathcal{Q}) = 0.5$, quand $(.)$ est l'application identité.

Ainsi, à chaque fois que le système pose une question au décideur, demandant de comparer une alternative binaire $1A0, A \subseteq \mathcal{Q}$, à un profil constant $\Lambda = (\lambda, \ldots, \lambda) \in [0,1]^q$:

- si $1A0$ est préférée à Λ, alors la réponse du décideur se traduit par la contrainte $v(A) \geq \lambda$,
- si Λ est préférée à $1A0$, alors c'est la contrainte $v(A) \leq \lambda$ qui doit être imposée.

En conséquence, si l'ensemble $\mathcal{P}$ contient uniquement des données de la forme $(1A0, \Lambda)$ et/ou $(\Lambda, 1A0)$, représentant respectivement les préférences observées $1A0 \succsim \Lambda$ et $\Lambda \succsim 1A0$, alors l'équation (2.11) peut être remplacée par des contraintes de bornes sur les variables. Pour assurer la compatibilité du modèle avec l'ensemble $\mathcal{P}$, il suffit en effet de mettre à jour les bornes d'un intervalle $[l_A, u_A]$ à chaque fois qu'une donnée impliquant l'alternative $1A0$ est insérée dans $\mathcal{P}$; cet intervalle contient alors toutes les valeurs de $v(A)$ qui sont compatibles avec $\mathcal{P}$. Notons que nous avons nécessairement $[l_\emptyset, u_\emptyset] = [0,0]$ et $[l_\mathcal{Q}, u_\mathcal{Q}] = [1,1]$ par la contrainte de normalisation (cf. équation 2.9), et que nous avons initialement $[l_A, u_A] = [0,1]$ pour tout $A \subset \mathcal{Q}$ non vide. Par ailleurs, puisque $\Omega_{\mathcal{P} \cup \{(1A0, \Lambda)\}}$ est l'ensemble des capacités $v \in \Omega_\mathcal{P}$ telles que $v(A) \geq \lambda$ et comme les capacités sont des fonctions monotones (cf. définition 13), nous avons nécessairement $v(B) \geq \lambda$ pour tout $B \supseteq A$. De ce fait, nous avons :

$$\Omega_{\mathcal{P} \cup \{(1A0, \Lambda)\}} = \Omega_{\mathcal{P} \cup \{(1B0, \Lambda) : B \supseteq A\}}$$

Similairement, comme $\Omega_{\mathcal{P}\cup\{(\Lambda,1A0)\}}$ est constitué des capacités $v \in \Omega_{\mathcal{P}}$ telles que $v(A) \leq \lambda$, alors :

$$\Omega_{\mathcal{P}\cup\{(\Lambda,1A0)\}} = \Omega_{\mathcal{P}\cup\{(\Lambda,1B0):B\subseteq A\}}$$

Ainsi, après chaque question faisant intervenir une alternative binaire $1A0, A \subset \mathcal{Q}$, et un profil constant $\Lambda = (\lambda, \ldots, \lambda) \in [l_A, u_A]^q$, nous pouvons réaliser les mises à jour suivantes :

- si la donnée $(1A0, \Lambda)$ est insérée dans $\mathcal{P}$, alors $l_B = \max\{l_B, \lambda\}$ pour tout B tel que $A \subseteq B \subseteq \mathcal{Q}$,
- si la donnée $(\Lambda, 1A0)$ est insérée dans $\mathcal{P}$, alors $u_B = \min\{u_B, \lambda\}$ pour tout $B \subseteq A$.

Il est important de souligner que restreindre les interactions à ce type de questions est suffisant pour éliciter entièrement les préférences du décideur ; en effet, les contraintes induites par ces questions nous permettent d'approcher les valeurs $v(A), A \subseteq \mathcal{Q}$, aussi finement que souhaité. Par ailleurs, notons que, par construction, nous avons $l_A \leq l_B$ et $u_A \leq u_B$ pour tout $A \subset B$. Cette dernière observation va nous permettre d'utiliser le résultat suivant :

Proposition 4. *Soient $[l_A, u_A], A \subseteq \mathcal{Q}$, des intervalles définissant les contraintes de borne des variables $v_A, A \subseteq \mathcal{Q}$. Supposons que nous avons $l_A \leq l_B$ et $u_A \leq u_B$ pour tout $A \subset B$. Dans ce cas, pour tout $\mathcal{A} \subset 2^{\mathcal{Q}}$, le sous-réseau des contraintes de monotonie $v_A \leq v_B$, où $A, B \in \mathcal{A}$ et $A \subset B$, est* globalement cohérent *(terme utilisé par exemple dans [Dechter, 1992]), autrement dit toute instanciation partielle des variables dans $\{v_A : A \in \mathcal{A}\}$ satisfaisant les contraintes de borne et de monotonie peut être étendue en une instanciation de toutes les variables $v_A, A \subseteq \mathcal{Q}$, vérifiant les contraintes de borne et de monotonie.*

Démonstration. Supposons que les inégalités $l_A \leq l_B$ et $u_A \leq u_B$ sont vraies pour tous sous-ensembles $A \subset B \subseteq \mathcal{Q}$. Soit I une instanciation des variables dans $\{v_A : A \in \mathcal{A}\}$ où $\mathcal{A} \subset 2^{\mathcal{Q}}$. Supposons que cette instanciation partielle respecte les contraintes suivantes : $v_A \in [l_A, u_A]$ pour tout sous-ensemble $A \in \mathcal{A}$ (contraintes de borne) et $v_A \leq v_B$ pour tous sous-ensembles $A, B \in \mathcal{A}$, $A \subset B$ (contraintes de monotonie). Considérons à présent l'extension complète de I décrite ci-dessous :

$$\forall A \notin \mathcal{A},\ v_A = \max\{l_A, \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}\} \tag{2.18}$$

Nous supposons ici que la valeur $\max_{A' \in \mathcal{A}: A' \subset A} v_{A'}$ est égale à $-\infty$ lorsque l'ensemble $\{A' \in \mathcal{A} : A' \subset A\}$ est vide. L'objectif est de montrer que cette instanciation complète vérifie à la fois les contraintes de borne et de monotonie.

Contraintes de borne : il s'agit de montrer que nous avons $v_A \in [l_A, u_A]$ pour tout $A \subseteq \mathcal{Q}$. Notons que, pour les sous-ensembles $A \in \mathcal{A}$, le résultat découle directement de nos hypothèses sur I. Pour tout sous-ensemble $A \notin \mathcal{A}$, nous avons forcément $v_A \geq l_A$ par définition de l'extension de I (cf. l'équation (2.18)). Pour montrer que nous avons aussi $v_A \leq u_A$, rappelons que nous avons $u_{A'} \leq u_A$ pour tout $A' \subset A$. Nous obtenons alors :

$$v_A = \max\{l_A, \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}\} \leq \max\{u_A, \max_{A' \in \mathcal{A}: A' \subset A} u_{A'}\} \leq \max\{u_A, u_A\} = u_A$$

Contraintes de monotonie : il s'agit cette fois-ci de montrer que l'inégalité $v_A \leq v_B$ est vraie pour tous $A \subset B \subseteq \mathcal{Q}$. Pour ce faire, nous distinguons les quatre cas suivants :

- $A \in \mathcal{A}$ *et* $B \in \mathcal{A}$ *:* nous obtenons directement $v_A \leq v_B$ par définition de l'instanciation I.
- $A \in \mathcal{A}$ *et* $B \notin \mathcal{A}$ *:* puisque $A \in \{B' \in \mathcal{A} : B' \subset B\}$ par définition, alors :

$$v_A \leq \max_{B' \in \mathcal{A}: B' \subset B} v_{B'} \leq \max\{l_B, \max_{B' \in \mathcal{A}: B' \subset B} v_{B'}\} = v_B$$

- $A \notin \mathcal{A}$ *et* $B \in \mathcal{A}$ *:* considérons un sous-ensemble $A' \subset A$ avec $A' \in \mathcal{A}$ (si un tel sous-ensemble existe). Puisque $A \subset B$, nous avons aussi $A' \subset B$. Comme A' et B sont deux éléments de $\mathcal{A}$, nous déduisons $v_{A'} \leq v_B$ par définition de I. Pour obtenir le résultat désiré, il suffit finalement d'écrire l'expression de v_A selon l'équation (2.18) :

$$v_A = \max\{l_A, \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}\} \leq \max\{l_B, v_B\} = v_B$$

- $A \notin \mathcal{A}$ *et* $B \notin \mathcal{A}$ *:* puisque $A \subset B$, nous avons $l_A \leq l_B$ par hypothèse. Par ailleurs, comme $A \subset B$, nous avons aussi $\{A' \in \mathcal{A} : A' \subset A\} \subseteq \{B' \in \mathcal{A} : B' \subset B\}$. Par conséquent, en utilisant l'équation (2.18), nous obtenons :

$$v_A = \max\{l_A, \max_{A' \in \mathcal{A}: A' \subset A} v_{A'}] \leq \max\{l_B, \max_{B' \in \mathcal{A}: B' \subset B} v_{B'}\} = v_B$$

Ainsi, nous avons construit une instanciation de toutes les variables qui étend l'instanciation I tout en satisfaisant les contraintes de borne et de monotonie. □

Puisque les sous-ensembles de $\mathcal{A}_{(x,y)}$ sont les seuls qui apparaissent dans la fonction objectif du programme (2.8)-(2.11), alors en appliquant la proposition 4 à l'ensemble $\mathcal{A} = \mathcal{A}_{(x,y)}$, nous savons que $\text{PMR}(x, y, \Omega_{\mathcal{P}})$ peut être obtenu en considérant uniquement les contraintes de monotonie impliquant les variables v_A, $A \in \mathcal{A}_{(x,y)}$. Ce résultat nous permet de calculer $\text{PMR}(x, y, \Omega_{\mathcal{P}})$ de manière exacte en utilisant un solveur pour résoudre le programme linéaire (plus simple) suivant :

$$(\text{LP}_1) \quad \begin{aligned}
\max_{v} \quad & \sum_{j=1}^{q} \Big(\big[y_{(j)} - y_{(j-1)}\big] v_{Y_{(j)}} - \big[x_{(j)} - x_{(j-1)}\big] v_{X_{(j)}} \Big) && (2.19)\\
s.c. \quad & v_{X_{(j+1)}} \leq v_{X_{(j)}}, \ \forall j \in \{1, \ldots, q-1\} && (2.20)\\
& v_{Y_{(j+1)}} \leq v_{Y_{(j)}}, \ \forall j \in \{1, \ldots, q-1\} && (2.21)\\
& v_{X_{(j)}} \leq v_{Y_{(k)}}, \ \forall j, k \in \mathcal{Q} : X_{(j)} \subset Y_{(k)} && (2.22)\\
& v_{Y_{(j)}} \leq v_{X_{(k)}}, \ \forall j, k \in \mathcal{Q} : Y_{(j)} \subset X_{(k)} && (2.23)\\
& l_{X_{(j)}} \leq v_{X_{(j)}} \leq u_{X_{(j)}}, \ \forall j \in \mathcal{Q} && (2.24)\\
& l_{Y_{(j)}} \leq v_{Y_{(j)}} \leq u_{Y_{(j)}}, \ \forall j \in \mathcal{Q} && (2.25)
\end{aligned}$$

où les équations (2.20-2.23) et les équations (2.24-2.25) représentent respectivement les nouvelles contraintes

de monotonie et de préférences. Le programme LP_1 comprend au plus $2q-1$ variables (au lieu de 2^q) et seulement un nombre quadratique de contraintes (au lieu d'un nombre exponentiel).

Le programme linéaire LP_1 peut en réalité être davantage simplifié, de sorte à obtenir une formulation ne faisant intervenir qu'un nombre linéaire de contraintes. Pour le montrer, nous notons c_A le coefficient de la variable de décision v_A, $A \in \mathcal{A}_{(x,y)}$, dans la fonction objectif du programme LP_1. Cette fonction peut donc s'écrire $\sum_{A\in\mathcal{A}_{(x,y)}} c_A v_A$ où :

$$
\begin{aligned}
&c_A = -(x_{(j)} - x_{(j-1)}) \leq 0, \text{ si } A = X_{(j)} \text{ et } A \neq Y_{(j)} \text{ pour un certain } j \in \mathcal{Q},\\
&c_A = y_{(j)} - y_{(j-1)} \geq 0, \text{ si } A \neq X_{(j)} \text{ et } A = Y_{(j)} \text{ pour un certain } j \in \mathcal{Q},\\
&c_A = y_{(j)} - y_{(j-1)} - (x_{(j)} - x_{(j-1)}), \text{ si } A = X_{(j)} \text{ et } A = Y_{(j)} \text{ pour un certain } j \in \mathcal{Q}.
\end{aligned}
$$

Remarquons que nous ne pouvons pas avoir $X_{(j)} = Y_{(k)}$ avec $j \neq k$ puisque ces deux ensembles de niveau auraient dans ce cas des cardinalités différentes. Notons par ailleurs que toutes les variables $v_A, A \in \mathcal{A}_{(x,y)}$, telles que $c_A = 0$ n'ont en réalité aucun impact sur la fonction objectif (2.19). Par conséquent, en utilisant la proposition 4, nous pouvons retirer toutes ces variables du programme linéaire pour obtenir une formulation plus simple. Cependant, pour alléger la présentation des résultats qui vont suivre, nous supposons à présent que $c_A \neq 0$ pour tout $A \in \mathcal{A}_{(x,y)}$. [1]

Tout d'abord, nous souhaitons montrer que l'équation (2.22) peut être enlevée du programme LP_1 car celle-ci engendre des redondances. Notons LP'_1 le programme linéaire obtenu en retirant cette équation du programme LP_1. La proposition suivante va nous permettre d'établir le résultat désiré :

Proposition 5. *Soit v une solution optimale de LP'_1. Pour tout $A \in \mathcal{A}_{(x,y)}$ tel que $c_A > 0$, nous avons :*

$$v_A = \min\{u_A, \min_{A'\in Pa(A)} v_{A'}\} \tag{2.26}$$

où $Pa(A) = \{A' \in \mathcal{A}_{(x,y)} : A' \supset A \text{ et } c_{A'} < 0\}$ est l'ensemble des "parents négatifs" de l'ensemble A.

Démonstration. Soit v^* une solution optimale du programme LP'_1. Supposons que v^* ne vérifie pas l'équation (2.26) pour au moins un ensemble $A \in \mathcal{A}_{(x,y)}$ tel que $c_A > 0$. Soit $\hat{v}$ la solution définie par :

- $\hat{v}_A = v^*_A$ pour tout $A \in \mathcal{A}_{(x,y)}$ tel que $c_A < 0$.
- $\hat{v}_A = \min\{u_A, \min_{A'\in Pa(A)} \hat{v}_{A'}\}$ pour tout $A \in \mathcal{A}_{(x,y)}$ tel que $c_A > 0$.

Par construction, la solution $\hat{v}$ vérifie l'équation (2.26) pour tout $A \in \mathcal{A}_{(x,y)}$ tel que $c_A > 0$. Notre objectif est de montrer que $\hat{v}$ est aussi une solution réalisable du programme LP'_1 et que $\hat{v}$ est strictement meilleure que la solution v^* (ce qui soulève une contradiction).

Pour prouver que $\hat{v}$ est une solution réalisable de LP'_1, nous devons montrer que la solution $\hat{v}$ vérifie les équations (2.20), (2.21), (2.23), (2.24) et (2.25) :

1. Dans le cas contraire, tous les résultats établis dans cette sous-section peuvent être obtenus en restreignant $\mathcal{A}_{(x,y)}$ aux ensembles A tels que $c_A \neq 0$ et en utilisant les mêmes arguments sur cette version simplifiée du programme.

- *Équation (2.20) :* soient $A, B \in \mathcal{A}_{(x,y)}$ tels que $A \subset B$, $c_A < 0$ et $c_B < 0$. Nous voulons montrer que l'inégalité $\hat{v}_A \leq \hat{v}_B$ est forcément vraie. Puisque $A \subset B$ et v^* est une solution réalisable de LP'_1, alors nous avons $v^*_A \leq v^*_B$ par l'équation (2.20). Comme par ailleurs nous $\hat{v}_A = v^*_A$ et $\hat{v}_B = v^*_B$ par définition de $\hat{v}$, alors nous avons bien $\hat{v}_A \leq \hat{v}_B$.
- *Équation (2.21) :* soient $A, B \in \mathcal{A}_{(x,y)}$ tels que $A \subset B$, $c_A > 0$ et $c_B > 0$. Il s'agit de montrer que l'inégalité $\hat{v}_A \leq \hat{v}_B$ est vérifiée. Comme $A \subset B$, alors nous avons $u_A \leq u_B$ par hypothèse et $Pa(B) \subseteq Pa(A)$ par définition. Par ailleurs, puisque $\hat{v}$ vérifie l'équation (2.26) pour tout $X \in \mathcal{A}_{(x,y)}$ tel que $c_X > 0$, alors nous avons :

$$\hat{v}_A = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\} \leq \min\{u_B, \min_{B' \in Pa(B)} \hat{v}_{B'}\} = \hat{v}_B$$

- *Équation (2.23) :* soient $A, B \in \mathcal{A}_{(x,y)}$ tels que $A \subset B$, $c_A > 0$ et $c_B < 0$. Montrons que $\hat{v}_A \leq \hat{v}_B$ est vraie. Comme $B \supset A$, alors $B \in Pa(A)$ par définition de $Pa(A)$. En utilisant l'équation (2.26), nous obtenons donc :

$$\hat{v}_A = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\} \leq \min\{u_A, \hat{v}_B\} \leq \hat{v}_B$$

- *Équations (2.24) et (2.25) :* nous devons montrer que les inégalités $l_A \leq \hat{v}_A \leq u_A$ sont vraies pour tout $A \in \mathcal{A}_{(x,y)}$. Remarquons que, pour tout $A \in \mathcal{A}_{(x,y)}$ tel que $c_A < 0$, nous avons nécessairement $l_A \leq \hat{v}_A \leq u_A$ puisque $\hat{v}_A = v^*_A$ et v^* est une solution réalisable. Soit $A \in \mathcal{A}_{(x,y)}$ tel que $c_A > 0$. Par définition, nous avons $\hat{v}_A \leq u_A$ (cf. équation (2.26)) est vraie. Par conséquent, nous avons juste à montrer que $\hat{v}_A \geq l_A$. Puisque $l_{A'} \geq l_A$ pour tout $A' \in Pa(A)$, nous obtenons finalement :

$$\hat{v}_A = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\} \geq \min\{l_A, \min_{A' \in Pa(A)} l_{A'}\} \geq l_A$$

Ainsi, $\hat{v}$ est bien une solution réalisable du programme LP'_1. Montrons maintenant que $\hat{v}$ est strictement meilleure que v^*. En d'autres termes, il s'agit de démontrer que l'inégalité suivante est vérifiée :

$$\sum_{A \in \mathcal{A}_{(x,y)}} c_A \hat{v}_A > \sum_{A \in \mathcal{A}_{(x,y)}} c_A v^*_A$$

Comme $v^*_A = \hat{v}_A$ pour tout $A \in \mathcal{A}_{(x,y)}$ tel que $c_A < 0$, alors il suffit de montrer :

- $v^*_A \leq \hat{v}_A$ pour tout $A \in \mathcal{A}_{(x,y)}$ tel que $c_A > 0$,
- $v^*_A < \hat{v}_A$ pour au moins un ensemble $A \in \mathcal{A}_{(x,y)}$ tel que $c_A > 0$.

Notons que nous avons forcément $v^*_A \leq u_A$ (par l'équation (2.25)) et $v^*_A \leq \min_{A' \in Pa(A)} v^*_{A'}$ (par l'équation (2.23)) puisque v^* est une solution réalisable ; par conséquent, nous avons $v^*_A \leq \min\{u_A, \min_{A' \in Pa(A)} v^*_{A'}\}$. Comme par ailleurs $v^*_{A'} = \hat{v}_{A'}$ pour tout $A' \in Pa(A)$, alors nous obtenons :

$$v^*_A \leq \min\{u_A, \min_{A' \in Pa(A)} v^*_{A'}\} = \min\{u_A, \min_{A' \in Pa(A)} \hat{v}_{A'}\} = \hat{v}_A$$

Observons que la première inégalité est stricte si l'équation (2.26) n'est pas vérifiée. Comme il existe bel et bien un sous-ensemble $A \in \mathcal{A}_{(x,y)}, c_A > 0$, tel que cette équation n'est pas satisfaite (par définition de v^*), alors nous concluons que la solution $\hat{v}$ est strictement meilleure que la solution v^* ; cette conclusion contredit nos hypothèses initiales. □

Cette proposition, donnant une condition nécessaire d'optimalité, nous permet de démontrer le résultat suivant :

Proposition 6. *Toute solution optimale de* LP'_1 *est une solution optimale de* LP_1.

Démonstration. Soit v une solution optimale du programme LP'_1. Nous devons montrer que la solution v vérifie l'équation (2.22) : il s'agit de prouver que nous avons nécessairement $v_{X_{(j)}} \leq v_{Y_{(k)}}$ pour tous $j, k \in \mathcal{Q}$ tels que $X_{(j)} \subset Y_{(k)}$. Puisque $c_{Y_{(k)}} > 0$, alors $v_{Y_{(k)}} = \min\{u_{v_{Y_{(k)}}}, \min_{A \in Pa(Y_{(k)})} v_A\}$ d'après la proposition 5. Nous distinguons alors les deux cas suivants :

- $v_{Y_{(k)}} = u_{Y_{(k)}}$ *:* dans ce cas, puisque $X_{(j)} \subset Y_{(k)}$, alors $u_{X_{(j)}} \leq u_{Y_{(k)}}$. Par ailleurs, nous avons $v_{X_{(j)}} \leq u_{X_{(j)}}$ par l'équation (2.24). Par conséquent :

$$v_{X_{(j)}} \leq u_{X_{(j)}} \leq u_{Y_{(k)}} = v_{Y_{(k)}}$$

- $v_{Y_{(k)}} = \min_{A \in Pa(Y_{(k)})} v_A$ *:* si $Pa(Y_{(k)}) = \emptyset$, alors $v_{Y_{(k)}} = u_{Y_{(k)}}$ et donc le résultat peut être obtenu comme dans le cas précédent. Dans le cas contraire, remarquons que nous avons forcément $Pa(Y_{(k)}) \subseteq \{A \in \mathcal{A}_{(x,y)} : c_A < 0\} \subseteq \{X_{(l)} : l \in \mathcal{Q}\}$. Par conséquent, il existe $l \in \mathcal{Q}$ tel que $v_{Y_{(k)}} = \min_{A \in Pa(Y_{(k)})} v_A = v_{X_{(l)}}$. Par ailleurs, puisque $X_{(j)} \subset Y_{(k)}$ (par hypothèse) et que $Y_{(k)} \subset X_{(l)}$ (par définition de $Pa(Y_{(k)})$), alors nous avons $X_{(j)} \subset X_{(l)}$ par transitivité de l'inclusion. Enfin, comme v vérifie l'équation (2.20), alors nous en déduisons :

$$v_{X_{(j)}} \leq v_{X_{(l)}} = v_{Y_{(k)}}$$

□

Cette proposition nous permet de conclure que l'équation (2.22) n'est pas nécessaire pour déterminer la solution optimale du programme LP_1. En réalité, nous pouvons encore retirer des contraintes de ce programme linéaire. Plus précisément, certaines contraintes correspondant à l'équation (2.23) ne sont pas utiles. En effet, s'il existe $j, k \in \mathcal{Q}$ tels que $Y_{(j)} \subset X_{(k)}$, alors nous avons aussi $Y_{(j)} \subset X_{(l)}$ pour tout $l \in \{1, \ldots, k-1\}$ car les ensembles de niveau de x forment une séquence emboîtée. Par conséquent, il existe des redondances entre les équations (2.23) et (2.20). Pour les éviter, il suffit d'imposer $v_{Y_{(j)}} \leq v_{X_{(k)}}$ uniquement lorsque $Y_{(j)} \subset X_{(k)}$ et $Y_{(j)} \not\subseteq X_{(k+1)}$. Par ailleurs, nous avons $Y_{(l)} \subset X_{(k)}$ pour tout $l \in \{j+1, \ldots, q\}$, car les ensembles de niveau de y forment eux aussi une séquence emboîtée ; ceci provoque une redondance avec l'équation (2.21) cette fois-ci. En conséquence, il convient d'imposer $v_{Y_{(j)}} \leq v_{X_{(k)}}$ uniquement lorsque $Y_{(j)} \subset X_{(k)}$, $Y_{(j)} \not\subseteq X_{(k+1)}$ et $Y_{(j-1)} \not\subseteq X_{(k)}$. Toutes ces simplifications

conduisent finalement au programme suivant :

$$\max_{v} \quad \sum_{j=1}^{q} \Big(\big[y_{(j)} - y_{(j-1)}\big]\, v_{Y_{(j)}} - \big[x_{(j)} - x_{(j-1)}\big]\, v_{X_{(j)}} \Big) \tag{2.27}$$

$$s.c. \quad v_{X_{(j+1)}} \leq v_{X_{(j)}},\ \forall j \in \{1,\dots,q-1\} \tag{2.28}$$

$$(\mathrm{LP}_2) \qquad v_{Y_{(j+1)}} \leq v_{Y_{(j)}},\ \forall j \in \{1,\dots,q-1\} \tag{2.29}$$

$$v_{Y_{(j)}} \leq v_{X_{(k)}},\ \forall j,k \in \mathcal{Q} : Y_{(j)} \subset X_{(k)},\ Y_{(j)} \not\subseteq X_{(k+1)},\ Y_{(j-1)} \not\subseteq X_{(k)} \tag{2.30}$$

$$l_{X_{(j)}} \leq v_{X_{(j)}} \leq u_{X_{(j)}},\ \forall j \in \mathcal{Q} \tag{2.31}$$

$$l_{Y_{(j)}} \leq v_{Y_{(j)}} \leq u_{Y_{(j)}},\ \forall j \in \mathcal{Q} \tag{2.32}$$

Cette formulation linéaire implique au plus $2q-1$ variables (associées aux éléments de $\mathcal{A}_{(x,y)}$) reliées par au plus $3(q-1)$ contraintes de monotonie (cf. équations (2.28)-(2.30)).

Exemple 17. *Considérons un problème de décision avec cinq critères (c'est-à-dire, $\mathcal{Q} = \{1,2,3,4,5\}$), et deux alternatives $x = (1, 0.8, 0.4, 0.5, 0.1)$ et $y = (0.8, 1, 0.6, 0.2, 0.4)$. Nous avons ici :*

$$\mathrm{Ch}(x,v) = 0.1v(\mathcal{Q}) + 0.3v(\{1,2,3,4\}) + 0.1v(\{1,2,4\}) + 0.3v(\{1,2\}) + 0.2v(\{1\})$$
$$\mathrm{Ch}(y,v) = 0.2v(\mathcal{Q}) + 0.2v(\{1,2,3,5\}) + 0.2v(\{1,2,3\}) + 0.2v(\{1,2\}) + 0.2v(\{2\})$$

Le programme linéaire LP_2 *associé au calcul de* $\mathrm{PMR}(x,y,\Omega_{\mathcal{P}})$ *est donc le suivant :*

$$\max_{v} \quad 0.2(v_{\{1,2,3,5\}} + v_{\{1,2,3\}} + v_{\{2\}} - v_{\{1\}}) + 0.1(v_{\mathcal{Q}} - v_{\{1,2\}} - v_{\{1,2,4\}}) - 0.3v_{\{1,2,3,4\}}$$

$$s.c. \quad v_{\{1\}} \leq v_{\{1,2\}} \leq v_{\{1,2,4\}} \leq v_{\{1,2,3,4\}} \leq v_{\mathcal{Q}} \tag{2.33}$$

$$v_{\{2\}} \leq v_{\{1,2\}} \leq v_{\{1,2,3\}} \leq v_{\{1,2,3,5\}} \leq v_{\mathcal{Q}} \tag{2.34}$$

$$v_{\{1,2,3\}} \leq v_{\{1,2,3,4\}} \tag{2.35}$$

$$l_{\{1\}} \leq v_{\{1\}} \leq u_{\{1\}},\ l_{\{1,2\}} \leq v_{\{1,2\}} \leq u_{\{1,2\}},\ l_{\{1,2,4\}} \leq v_{\{1,2,4\}} \leq u_{\{1,2,4\}}$$
$$l_{\{1,2,3,4\}} \leq v_{\{1,2,3,4\}} \leq u_{\{1,2,3,4\}},\ l_{\mathcal{Q}} \leq v_{\mathcal{Q}} \leq u_{\mathcal{Q}},\ l_{\{2\}} \leq v_{\{2\}} \leq u_{\{2\}}$$
$$l_{\{1,2,3\}} \leq v_{\{1,2,3\}} \leq u_{\{1,2,3\}},\ l_{\{1,2,3,5\}} \leq v_{\{1,2,3,5\}} \leq u_{\{1,2,3,5\}}$$

où les équations (2.33) et (2.34) correspondent respectivement aux contraintes des équations (2.28) et (2.29), tandis que l'équation (2.35) nous donne la seule contrainte associée à l'équation (2.30) ; toutes les autres contraintes proviennent des équations (2.31)-(2.32). Ainsi, nous avons besoin ici de seulement 8 variables (au lieu de 32) et 9 contraintes de monotonie (au lieu de 80).

2.2.2 Optimisation du critère Minimax Regret par un algorithme itératif

Bien que les solveurs actuels soient capables de résoudre relativement facilement toute instance du programme linéaire LP_2, la détermination du mMR requiert dans le pire cas un nombre quadratique

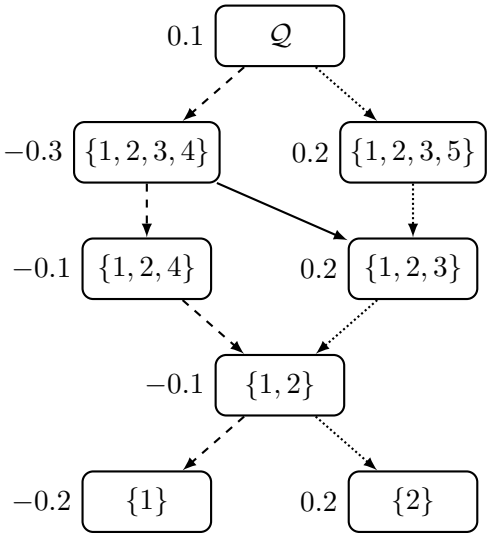
de telles résolutions (cf. définition 36). Dans l'optique de réduire les temps de calculs, nous proposons maintenant une méthode itérative permettant de résoudre toute instance du programme LP_2 sans avoir recours à la programmation linéaire. Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ le graphe de contraintes associé aux contraintes de monotonie du programme LP_2. Le graphe $\mathcal{G}$ est formellement défini comme suit :

- $\mathcal{V} = \{X_{(j)} : j \in \mathcal{Q}\} \cup \{Y_{(j)} : j \in \mathcal{Q}\}$ est l'ensemble des nœuds du graphe ; chaque nœud est ainsi associé à un sous-ensemble de critères $A \subseteq \mathcal{Q}$ tel que la variable de décision v_A est dans LP_2.
- $\mathcal{E}$ est l'ensemble des arcs du graphe ; l'arc (A, B) appartient au graphe si et seulement si la contrainte $v_A \geq v_B$ apparaît dans les équations (2.28)-(2.30).

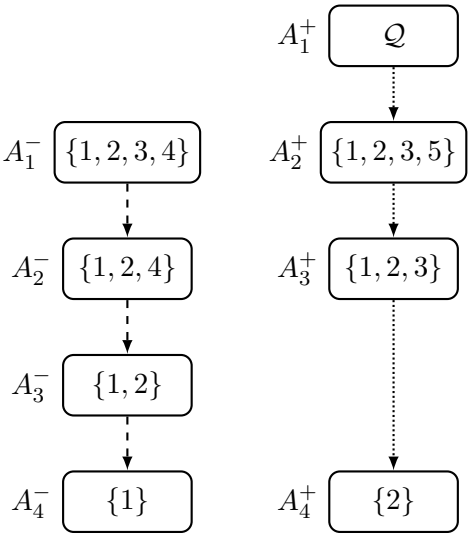
Dans le graphe $\mathcal{G}$, les arcs correspondant aux équations (2.28)-(2.29) forment deux chemins caractérisés par les listes de nœuds suivantes : $P_x = (X_{(1)}, \ldots, X_{(q)})$ et $P_y = (Y_{(1)}, \ldots, Y_{(q)})$. Notons que ces deux chemins passent à eux deux par tous les nœuds du graphe de contraintes. Soit A^- (resp. A^+) la sous-séquence de P_x (resp. P_y) composée de tous les nœuds A tels que $c_A < 0$ (resp. $c_A > 0$), où c_A est le coefficient de la variable v_A dans la fonction objectif (2.27). Par définition, les séquences A^- et A^+ vérifient les propriétés suivantes :

- A^- et A^+ incluent, à elles deux, tous les nœuds de $\mathcal{V}$ et n'ont aucun nœud en commun.
- $A^+_j \supset A^+_{j+1}$ pour tout $j \in \{1, \ldots, |A^+| - 1\}$, où A^+_k représente le $k^{\text{ème}}$ nœud dans la séquence A^+.
- $A^-_j \supset A^-_{j+1}$ pour tout $j \in \{1, \ldots, |A^-| - 1\}$, où A^-_k représente le $k^{\text{ème}}$ nœud dans la séquence A^-.
- pour tout $j \in \{1, \ldots, |A^-|\}$, il existe au plus un indice $k \in \{1, \ldots, |A^+|\}$ tel que $(A^-_j, A^+_k) \in \mathcal{E}$.

Dans la suite, les variables correspondant aux nœuds dans A^- (resp. A^+) seront appelées *variables négatives* (resp. *variables positives*) puisque celles-ci impactent négativement (resp. positivement) la fonction objectif. À titre d'exemple, nous donnons ci-après le graphe de contraintes associé à l'exemple 17 :



(a) Graphe de contraintes associé au calcul de $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}})$; les arcs tiretés (resp. en pointillés) représentent le chemin P_x (resp. P_y) ; la valeur c_A est indiquée à gauche de chaque nœud A.



(b) Séquences A^- et A^+ associées au calcul de $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}})$; dans cette figure, les éléments successifs de la séquence A^- (resp. A^+) sont reliés entre eux par des arcs tiretés (resp. en pointillés).

FIGURE 2.1 – Graphe de contraintes et les séquences A^+ et A^- correspondant à l'exemple 17.

Puisque la séquence A^+ (resp. A^-) contient tous les nœuds A tel que la variable v_A impacte positivement (resp. négativement) la fonction objectif, nous souhaitons en réalité maximiser (resp. minimiser) la variable v_A pour tout $A \in A^+$ (resp. pour tout $A \in A^-$) afin de maximiser la fonction objectif. De ce fait, s'il n'existe aucun arc de type (A_j^-, A_k^+) dans $\mathcal{E}$ (représentant la contrainte $v_{A_j^-} \geq v_{A_k^+}$), alors la solution optimale du programme LP_2 peut être obtenue aisément ; il suffit en effet de mettre v_A à sa borne inférieure l_A pour tout $A \in A^-$ et v_A à sa borne supérieure u_A pour tout $A \in A^+$. Dans le cas contraire, pour tout arc de type (A_j^-, A_k^+) dans $\mathcal{E}$, il s'agit de déterminer si la variable $v_{A_j^-}$ doit être fixée à sa borne inférieure $l_{A_j^-}$ (au prix de contraindre la variable $v_{A_k^+}$) ou bien à une valeur plus grande. Remarquons que, pour tout $l \in \{k+1, \ldots, |A^+|\}$, la contrainte $v_{A_j^-} \geq v_{A_l^+}$ est indirectement imposée par les contraintes de monotonie car nous avons $A_l^+ \subset A_k^+$ par définition de la séquence A^+. Par conséquent, pour tout $j \in \{1, \ldots, |A^-|\}$, il s'agit de décider si la variable $v_{A_j^-}$ doit être fixée à sa borne inférieure $l_{A_j^-}$ ou bien à une valeur supérieure pour préserver les variables dans l'ensemble $\{v_{D_k^j} : 1 \leq k \leq |D^j|\}$, où D^j représente la sous-séquence de A^+ restreinte aux descendants de A_j^- dans le graphe $\mathcal{G}$ et D_k^j est le $k^{\text{ème}}$ élément de D^j.

Afin de déterminer la bonne décision, nous proposons de procéder de manière itérative. Le principe général est le suivant : tout d'abord, nous souhaitons savoir si la variable $v_{A_j^-}$ doit être mise à une valeur plus grande que $u_{D_1^j}$ (pour préserver la variable $v_{D_1^j}$) ou bien si la variable $v_{A_j^-}$ doit être instanciée à une valeur plus petite que $u_{D_1^j}$. Nous sommes dans le premier cas lorsque l'inégalité suivante est vérifiée :

$$\sum_{\substack{j \leq k \leq |A^-| \\ A_k^- \supset D_1^j}} |c_{A_k^-}| < c_{D_1^j} \tag{2.36}$$

En effet, cette inégalité nous informe que la variable $v_{D_1^j}$ a un plus grand impact sur la fonction objectif que la variable $v_{A_j^-}$; en d'autres termes, diminuer la valeur de la variable $v_{D_1^j}$ au profit de la variable $v_{A_j^-}$ conduit à une diminution locale de la valeur de la fonction objectif. Dans cette situation, la variable $v_{A_j^-}$ doit donc être instanciée à une valeur plus grande que $u_{D_1^j}$ pour éviter de contraindre la variable $v_{D_1^j}$ (qui joue un rôle plus important). Si en revanche l'équation (2.36) n'est pas vérifiée, alors une diminution de la valeur de $v_{D_1^j}$ au profit de la variable $v_{A_j^-}$ entraîne cette fois-ci une augmentation locale de la valeur de la fonction objectif. Dans ce cas, nous déduisons que la variable $v_{A_j^-}$ doit être instanciée à une valeur plus petite que $u_{D_1^j}$. Pour en connaître précisément la valeur, ce test doit être itéré sur le nœud D_2^j, et cætera. Si jamais ce test échoue pour tous les nœuds D_k^j, avec $1 \leq k \leq |D^j|$, alors la variable $v_{A_j^-}$ peut être mise à la valeur $l_{A_j^-}$ (sa borne inférieure). Remarquons que la procédure itérative que nous venons de décrire brièvement ne tient pas compte du fait que la variable $v_{A_j^-}$ ne doit pas être instanciée à une valeur plus petite que sa borne inférieure. Cette procédure itérative est présentée plus en détail dans l'algorithme 1, qui précise notamment comment les contraintes de borne doivent être gérées durant la détermination des valeurs des variables $v_{A_j^-}$, avec $j \in \{1, \ldots, |A^-|\}$. Notons que cet algorithme a une complexité en $O(q^2)$ car nous avons d'une part $|A^-| \leq q$ et d'autre part $|D^j| \leq |A^+| \leq q$ pour tout $j \in \{1, \ldots, |A^-|\}$.

Algorithm 1: Résolution du problème PMR$(x, y, \Omega_{\mathcal{P}})$

Input: deux alternatives $x, y \in X$, un ensemble de données $\mathcal{P}$
Output: une instanciation des variables v_A, $A \in \mathcal{A}_{(x,y)}$

1 Construction des séquences A^- et A^+ à partir de $\mathcal{G}$ le graphe de contraintes
2 **for** $A \in A^+$ **do** $v_A \leftarrow u_A$
3 **for** $j = 1 \dots |A^-|$ **do**
4 $\quad v_{A_j^-} \leftarrow l_{A_j^-}$
5 $\quad k \leftarrow 1$
6 $\quad c^+ \leftarrow 0$
7 $\quad c^- \leftarrow 0$
8 $\quad$ **while** $k \leq |D^j|$ *et* $v_{D_k^j} > l_{A_j^-}$ **do**
9 $\quad\quad c^+ \leftarrow c^+ + c_{D_k^j}$
10 $\quad\quad c^- \leftarrow c^- + \sum_{l \in \{j,\dots,|A^-|\} : D_1^l = D_k^j} c_{A_l^-}$
11 $\quad\quad$ **if** $c^+ + c^- > 0$ **then**
12 $\quad\quad\quad v_{A_j^-} \leftarrow v_{D_k^j}$
13 $\quad\quad\quad$ break
14 $\quad\quad$ **end**
15 $\quad\quad k \leftarrow k + 1$
16 $\quad$ **end**
17 $\quad$ **for** $k = 1 \dots |D^j|$ **do** $v_{D_k^j} \leftarrow \min\{v_{D_k^j}, v_{A_j^-}\}$
18 **end**
19 **return** v

La proposition suivante va nous être utile pour démontrer la validité de l'algorithme 1 :

Proposition 7. *À la fin de chaque itération j de la boucle "for" de l'algorithme 1, nous avons :*

$$\forall A \in A^+,\ v_A = \min\{u_A, \min_{A' \in Pa_j(A)} v_{A'}\}$$

où $Pa_j(A) = \{A_k^-, 1 \leq k \leq j : A \subset A_k^-\}$.

Démonstration. Montrons par récurrence que la proposition suivante, notée $P(j)$, est vraie à la fin de chaque itération $j \in \{0, \dots, |A^-|\}$:

$$P(j) : \forall A \in A^+,\ v_A = \min\{u_A, \min_{A' \in Pa_j(A)} v_{A'}\}$$

Pour l'itération $j = 0$ (avant entrer dans la boucle "for"), la proposition $P(j)$ est trivialement vraie puisque, pour tout $A \in A^+$, $Pa_j(A) = \emptyset$ et v_A est initialisée à u_A (cf. ligne 2). Supposons maintenant que $P(j-1)$ est vraie pour une certaine itération $j \in \{1, \dots, |A^-| - 1\}$. Nous voulons montrer que $P(j)$ est dans ce cas nécessairement vraie aussi. Soit $A \in A^+$. À la fin de l'itération $j-1$, nous avons $v_A = \min\{u_A, \min_{A' \in Pa_{j-1}(A)} v_{A'}\}$ par hypothèse de récurrence ; notons qu'aucune variable de l'ensemble

$\{v_{A'} : A' \in Pa_{j-1}(A)\}$ n'est ensuite modifiée durant l'itération j. Supposons tout d'abord que $A \notin A_j^-$. Dans ce cas, nous savons que la variable v_A n'est pas modifiée par l'algorithme durant l'itération j car seules les variables positives associées aux éléments de D^j sont modifiées. De plus, comme $Pa_j(A) = Pa_{j-1}(A)$, alors nous pouvons directement inférer le résultat : $v_A = \min\{u_A, \min\limits_{A' \in Pa_j(A)} v_{A'}\}$.

Supposons maintenant que $A \subset A_j^-$. Dans ce cas, durant l'itération j, la variable v_A est modifiée selon la formule donnée en ligne 17 : $v_A = \min\{v_A, v_{A_j^-}\}$. Ainsi, à la fin de l'itération j, nous avons :

$$\begin{aligned}
v_A &= \min\{v_A, v_{A_j^-}\} \text{ par la ligne 17} \\
&= \min\{\min\{u_A, \min_{A' \in Pa_{j-1}(A)} v_{A'}\}, v_{A_j^-}\} \text{ par hypothèse de récurrence} \\
&= \min\{u_A, \min_{A' \in Pa_{j-1}(A)} v_{A'}, v_{A_j^-}\} \\
&= \min\{u_A, \min_{A' \in Pa_{j-1}(A) \cup \{A_j^-\}} v_{A'}\} \\
&= \min\{u_A, \min_{A' \in Pa_j(A)} v_{A'}\}
\end{aligned}$$

Finalement, nous avons montré que $P(j)$ est vérifiée lorsque $P(j-1)$ est vraie. Par conséquent, pour tout $j \in \{0, \ldots, |A^-|\}$, nous avons bien $v_A = \min\{u_A, \min_{A' \in Pa_j(A)} v_{A'}\}$ pour tout $A \in A^+$. □

Ce résultat nous permet de dériver la proposition suivante :

Proposition 8. *L'algorithme 1 retourne une solution v vérifiant l'équation (2.26) pour tout $A \in A^+$.*

Démonstration. Le résultat découle directement de la proposition 7 (avec $j = |A^-|$) puisque $Pa_{|A^-|}(A) = Pa(A)$ par définition. □

Cette proposition nous informe que l'algorithme 1 construit une solution vérifiant les conditions nécessaires d'optimalité données par l'équation (2.26). Ces conditions nous ont en réalité fourni une manière opérationnelle de déterminer les valeurs optimales des variables positives, lorsque celles des variables négatives ont été spécifiées avant. Pour montrer que l'algorithme 1 est valide, il suffit maintenant de montrer que les variables négatives sont "correctement" instanciées par cet algorithme. Ceci est réalisé dans la preuve de la proposition suivante (donnée en annexe) :

Proposition 9. *L'algorithme 1 retourne une solution optimale du programme LP_2.*

Pour illustrer le fonctionnement de notre algorithme, nous détaillons ci-après son exécution sur le problème de l'exemple 17 :

Exemple 18. *Reprenons le problème décrit dans l'exemple 17, avec les alternatives x et y définies par $x = (1, 0.8, 0.4, 0.5, 0.1)$ et $y = (0.8, 1, 0.6, 0.2, 0.4)$. Nous souhaitons calculer* $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}})$ *en appliquant l'algorithme 1. Rappelons que le programme linéaire LP_2 correspondant à ce problème d'optimisation est donné dans l'exemple 17. Par ailleurs, le graphe des contraintes associé à ce problème, ainsi que les séquences A^+ et A^- de ce graphe, sont quant à eux représentés dans la figure 2.1.*

Supposons que le décideur a déjà répondu à quelques questions et que les intervalles courants $[l_A, u_A]$, $A \in \mathcal{A}_{(x,y)}$, *sont les suivants : Déroulons l'algorithme 1 sur cet exemple. Tout d'abord, la ligne 2 nous*

$v_{\{1\}}$	$v_{\{1,2\}}$	$v_{\{1,2,4\}}$	$v_{\{1,2,3,4\}}$	$v_{\{2\}}$	$v_{\{1,2,3\}}$	$v_{\{1,2,3,5\}}$	$v_{\mathcal{Q}}$
$[0, 0.5]$	$[0.2, 0.7]$	$[0.3, 0.9]$	$[0.6, 1]$	$[0.1, 0.4]$	$[0.4, 0.8]$	$[0.5, 0.9]$	$[1, 1]$

dicte d'initialiser la variable v_A *à sa borne supérieure* u_A *pour tout ensemble* $A \in A^+$ *:*

$$v_{\mathcal{Q}} = 1,\ v_{\{1,2,3,5\}} = 0.9,\ v_{\{1,2,3\}} = 0.8 \ \textit{ et } \ v_{\{2\}} = 0.4$$

La première itération de la boucle "for" se concentre sur le nœud $A_1^- = \{1, 2, 3, 4\}$ *ainsi que sur la séquence de ses descendants* $D^1 = (\{1, 2, 3\}, \{2\})$. *Tout d'abord, la variable* $v_{\{1,2,3,4\}}$ *est instanciée à sa borne inférieure (cf. ligne 4) :*

$$v_{\{1,2,3,4\}} = l_{\{1,2,3,4\}} = 0.6$$

Puis, durant la première itération de la boucle "while", c'est le descendant $D_1^1 = \{1, 2, 3\}$ *qui est étudié. Puisque* $v_{\{1,2,3\}} = 0.8 > 0.6 = l_{\{1,2,3,4\}}$, *alors les valeurs* $c^+ = c_{\{1,2,3\}} = 0.2$ *(cf. ligne 9) et* $c^- = c_{\{1,2,3,4\}} = -0.3$ *(cf. ligne 10) sont calculées. Cette itération de la boucle "while" s'arrête ensuite puisque la condition* $c^+ + c^- > 0$ *n'est pas vérifiée. C'est alors le descendant* $D_2^1 = \{2\}$ *qui est étudié à l'itération suivante. Puisque* $v_{\{2\}} = 0.4 \leq 0.6 = l_{\{1,2,3,4\}}$, *alors la boucle "while" s'arrête immédiatement. Cette itération de la boucle "for" se termine alors en réalisant les mises à jour suivantes (cf. ligne 17) :*

$$v_{\{1,2,3\}} = \min\{v_{\{1,2,3\}}, v_{\{1,2,3,4\}}\} = \min\{0.8, 0.6\} = 0.6 \ \textit{ et } \ v_{\{2\}} = \min\{v_{\{2\}},\ v_{\{1,2,3,4\}}\} = \min\{0.4, 0.6\} = 0.4$$

La seconde itération de la boucle "for" traite le nœud $A_2^- = \{1, 2, 4\}$ *dont la séquence de descendants est* $D^2 = (\{2\})$. *La variable* $v_{\{1,2,4\}}$ *est tout d'abord fixée à sa valeur minimale (cf. ligne 4) :*

$$v_{\{1,2,4\}} = l_{\{1,2,4\}} = 0.3$$

Ensuite, la première itération de la boucle "while" s'intéresse au nœud $D_1^2 = \{2\}$. *Comme* $v_{\{2\}} = 0.4 > 0.3 = l_{\{1,2,4\}}$, *alors les valeurs* $c^+ = c_{\{2\}} = 0.2$ *et* $c^- = c_{\{1,2,4\}} + c_{\{1,2\}} = -0.2$ *sont calculées d'après les lignes 9 et 10. Puisque la condition* $c^+ + c^- > 0$ *n'est pas vérifiée à cette itération, alors celle-ci se termine immédiatement. Comme en plus* $|D^2| = 1$, *alors la boucle "while" s'achève en même temps. Finalement, l'algorithme réalise la mise à jour suivante (cf. ligne 17) :*

$$v_{\{2\}} = \min\{v_{\{2\}}, v_{\{1,2,4\}}\} = \min\{0.4, 0.3\} = 0.3$$

Durant la troisième itération, c'est le nœud $A_3^- = \{1, 2\}$ *qui est étudié, la séquence de ses descendants étant* $D^3 = (\{2\})$. *Tout d'abord, la variable* $v_{\{1,2\}}$ *est instanciée à sa valeur minimale (cf. ligne 4) :*

$$v_{\{1,2\}} = l_{\{1,2\}} = 0.2$$

Puis, l'attention se porte sur le descendant $D_1^3 = \{2\}$. *Puisque* $v_{\{2\}} = 0.3 > 0.2 = l_{\{1,2\}}$, *alors les valeurs* $c^+ = c_{\{2\}} = 0.2$ *(cf. ligne 9) et* $c^- = c_{\{1,2\}} = -0.1$ *(cf. ligne 10) sont calculées. La condition* $c^+ + c^- > 0$ *est cette fois-ci vérifiée. Par conséquent, la valeur de* $v_{\{1,2\}}$ *est modifiée comme suit (cf. ligne 12) :*

$$v_{\{1,2\}} = v_{\{2\}} = 0.3$$

La boucle "while" s'interrompt alors immédiatement suite à l'exécution de l'instruction "break". Puis, la mise à jour suivante est effectuée par l'algorithme (cf. ligne 17) :

$$v_{\{2\}} = \min\{v_{\{2\}}, v_{\{1,2\}}\} = \min\{0.3, 0.3\} = 0.3$$

À la dernière itération de la boucle "for" est considéré le nœud $A_4^- = \{1\}$. *La variable* $v_{\{1\}}$ *est alors instanciée à sa valeur minimale en suivant la ligne 4 :*

$$v_{\{1\}} = l_{\{1\}} = 0$$

Cette itération s'arrête ensuite parce que nous avons ici $D^4 = \emptyset$. *En résumé, sur cet exemple, l'algorithme construit et retourne l'instanciation complète suivante :*

$v_{\{1\}}$	$v_{\{1,2\}}$	$v_{\{1,2,4\}}$	$v_{\{1,2,3,4\}}$	$v_{\{2\}}$	$v_{\{1,2,3\}}$	$v_{\{1,2,3,5\}}$	$v_{\mathcal{Q}}$
0	0.3	0.3	0.6	0.3	0.6	0.9	1

Dans cette sous-section, nous avons vu que le problème d'optimisation $\text{PMR}(x, y, \Omega_{\mathcal{P}})$ peut être résolu par un algorithme itératif de complexité quadratique, pourvu que les données de préférences disponibles soient de la forme $(1A0, \Lambda)$ ou $(\Lambda, 1A0)$. Dans le cadre de l'élicitation incrémentale, cette dernière condition nécessite de restreindre les interactions avec le décideur à un certain type de questions, ce qui exclut la possibilité d'utiliser la stratégie CSS (présentée en section 1.4.2) pour engendrer des questions de préférence. Il s'agit donc pour nous de concevoir une stratégie d'élicitation efficace, produisant uniquement des questions de comparaison entre des alternatives binaires de type $1A0$ et des profils constants de type $\Lambda = (\lambda, \ldots, \lambda)$; ceci est l'objet de la sous-section suivante.

2.2.3 Stratégie d'élicitation

La stratégie de génération de questions que nous proposons dans cette sous-section est une heuristique du critère de sélection WmMR (introduit en section 1.4.2). Pour pouvoir la présenter de manière formelle, plaçons-nous à une itération quelconque de la méthode d'élicitation et détaillons l'étape de sélection de la prochaine question à poser. Soient $[l_A, u_A]$, $A \subseteq \mathcal{Q}$, les intervalles représentant toutes les valeurs $v(A)$ compatibles avec l'ensemble $\mathcal{P}$ des informations collectées jusque là. Pour simplifier la présentation, nous adoptons ici la notation (A, λ) pour faire référence à la question demandant au décideur de comparer les alternatives $1A0$ et $\Lambda = (\lambda, \ldots, \lambda)$ avec $A \subseteq \mathcal{Q}$ et $\lambda \in [l_A, u_A]$.

Dans notre contexte, une question optimale pour le critère de sélection WmMR est caractérisée par une paire (A, λ) minimisant la quantité suivante (cf. définition 37) :

$$\mathrm{WmMR}((A,\lambda),\Omega_{\mathcal{P}}) = \max\Big\{\mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(1A0,\Lambda)\}}),\ \mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(\Lambda,1A0)\}})\Big\}$$

Poser une telle question permet par définition de réduire le plus possible la valeur mMR dans le pire scénario de réponse de décideur. Par définition du critère WmMR, pour déterminer une question optimale, il est nécessaire de calculer, pour tout $A \subseteq \mathcal{Q}$, la quantité suivante :

$$\begin{aligned}\lambda_A &= \arg\min_{\lambda\in[l_A,u_A]} \mathrm{WmMR}((A,\lambda),\Omega_{\mathcal{P}}) \\ &= \arg\min_{\lambda\in[l_A,u_A]} \max\Big\{\mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(1A0,\Lambda)\}}),\ \mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(\Lambda,1A0)\}})\Big\} \qquad (2.37)\end{aligned}$$

En effet, il faut ensuite choisir un ensemble A^* dans $\arg\min_{A\subseteq\mathcal{Q}} \mathrm{WmMR}((A,\lambda_A),\Omega_{\mathcal{P}})$ pour que la paire (A^*, λ_{A^*}) caractérise une question optimale pour le critère WmMR.

D'après l'équation (2.37), déterminer la valeur λ_A pour un ensemble $A \subseteq \mathcal{Q}$ donné revient à minimiser sur $\lambda \in [l_A, u_A]$ le maximum entre les valeurs $\mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(1A0,\Lambda)\}})$ et $\mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(\Lambda,1A0)\}})$. Observons que la valeur $\mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(1A0,\Lambda)\}})$ décroît quand la valeur λ augmente car la donnée $(1A0, \Lambda)$ induit les contraintes $v(B) \geq \lambda$, avec $B \supseteq A$, qui deviennent de plus en plus contraignantes quand λ augmente. Similairement, la fonction $\mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}\cup\{(\Lambda,1A0)\}})$ est croissante en λ car la donnée $(\Lambda, 1A0)$ impose les contraintes $v(B) \leq \lambda$, avec $B \subseteq A$, qui deviennent de moins en moins contraignantes quand λ croît. Comme par ailleurs ces deux fonctions ont la même valeur maximale (qui est $\mathrm{mMR}(\mathcal{X},\Omega_{\mathcal{P}})$), alors nous en concluons que celles-ci s'intersectent nécessairement. D'après l'équation (2.37), cette intersection caractérise exactement la valeur λ_A, qui peut donc être obtenue en utilisant un algorithme dichotomique classique (cf. figure 2.2 pour une illustration graphique).

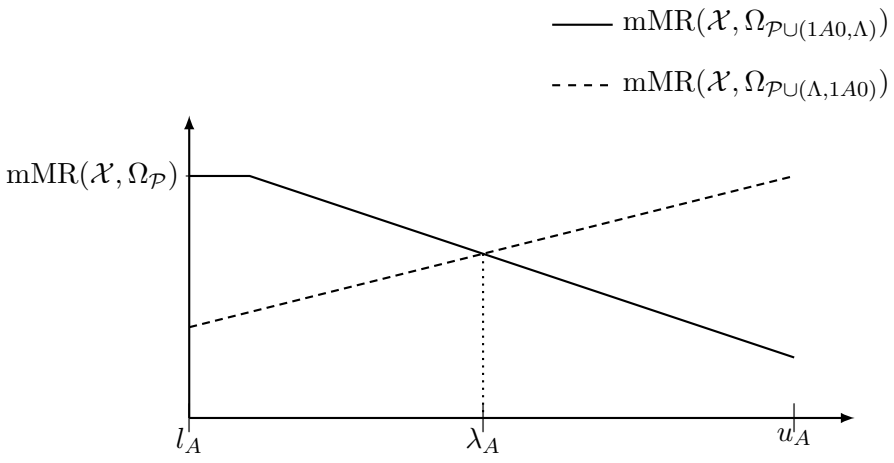


FIGURE 2.2 – Caractérisation de la valeur λ_A pour un ensemble $A \subseteq \mathcal{Q}$ donné.

Notons que si jamais l'égalité $\text{WmMR}((A^*, \lambda_{A^*}), \Omega_{\mathcal{P}}) = \text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})$ est vraie à l'itération courante, alors rien ne garantit que poser la question (A^*, λ_{A^*}) conduise à une réduction de la valeur mMR. Dans ce cas, un critère de sélection moins conservateur que le WmMR peut être envisagé pour discriminer entre les différentes questions possibles. Par exemple, nous pouvons choisir la question de type $(A, \frac{l_A+u_A}{2})$ qui minimise la moyenne des valeurs mMR obtenus dans les différents scénarios de réponses.

Par ailleurs, soulignons le fait que la détermination d'une question optimale pour le critère WmMR nécessite de considérer les $2^q - 2$ sous-ensembles propres de $\mathcal{Q}$, ce qui n'est pas envisageable lorsque le nombre de critères est important. Pour que cette étape de sélection soit réalisable en pratique, nous proposons de concentrer nos efforts sur des sous-ensembles A directement impliqués dans le calcul de la valeur $\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})$. L'idée est de contraindre davantage des variables induisant la valeur $\text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})$ courante pour que celle-ci soit potentiellement plus petite à l'itération suivante. L'heuristique que nous proposons se focalise plus précisément sur les sous-ensembles A intervenant dans le calcul de la valeur $\text{PMR}(x^*, y^*, \Omega_{\mathcal{P}})$, où x^* est une alternative optimale pour le critère de décision Minimax Regret et y^* est une alternative arbitrairement choisie dans l'ensemble $\arg\max_{y \in \mathcal{X}} \text{PMR}(x^*, y, \Omega_{\mathcal{P}})$; ce choix est essentiellement motivé par le fait que l'égalité $\text{PMR}(x^*, y^*, \Omega_{\mathcal{P}}) = \text{mMR}(\mathcal{X}, \Omega_{\mathcal{P}})$ est vraie par définition des alternatives x^* et y^*. Ainsi, les sous-ensembles A considérés par notre méthode sont tous ceux contenus dans l'ensemble suivant :

$$\mathcal{A}_{(x^*,y^*)} = \{X^*_{(j)} : j \in \mathcal{Q}\} \cup \{Y^*_{(j)} : j \in \mathcal{Q}\}$$

où $X^*_{(j)}$ et $Y^*_{(j)}$ représentent respectivement le $j^{\text{ème}}$ ensemble de niveau de x^* et y^* (cf. définition 15). Avec cette heuristique, au plus $2q - 2$ sous-ensembles A sont étudiés au lieu des $2^q - 2$ sous-ensembles propres de $\mathcal{Q}$; dans la sous-section suivante, nous verrons que notre stratégie de génération de questions est relativement efficace en pratique malgré cette restriction.

Enfin, il est important de remarquer que notre stratégie ne peut créer des incohérences dans l'ensemble $\mathcal{P}$ des données collectées sur les préférences du décideur ; autrement dit, l'ensemble $\Omega_{\mathcal{P}}$ ne peut devenir vide après insertion de la réponse du décideur dans l'ensemble $\mathcal{P}$. En effet, par définition des intervalles $[l_A, u_A]$, $A \subseteq \mathcal{Q}$, les deux réponses $1A0 \succsim (\lambda, \ldots, \lambda)$ et $(\lambda, \ldots, \lambda) \succsim 1A0$ sont tout à fait possibles si λ appartient à l'intervalle $[l_A, u_A]$. Plus précisément, celles-ci se traduisent respectivement par des contraintes de types "$v(B) \geq \lambda$" et "$v(B) \leq \lambda$" avec $\lambda \in [l_B, u_B]$, ce qui transforme les intervalles $[l_B, u_B]$ en $[\lambda, u_B]$ et $[l_B, \lambda]$ respectivement (qui sont bien non vides dans les deux cas).

2.2.4 Résultats expérimentaux

Les premiers tests que nous présentons ici ont pour but d'évaluer l'efficacité de notre stratégie de génération de questions présentée en Section 2.2.3 en terme de nombre de questions nécessaires pour pouvoir formuler une recommandation. Rappelons que chaque question engendrée par cette procédure implique deux alternatives fictives choisies soigneusement : une alternative de type $1A0$, $A \subseteq \mathcal{Q}$, et un profil d'utilité constant de type $\Lambda = (\lambda, \ldots, \lambda)$, $\lambda \in [l_A, u_A]$. Par conséquent, comme base de comparaison,

nous considérons la stratégie de génération de questions consistant à choisir, à chaque itération de la procédure d'élicitation, l'ensemble A et l'utilité $\lambda \in [l_A, u_A]$ aléatoirement. Partant d'une base de données $\mathcal{P}$ vide, nous simulons les réponses aux questions à l'aide d'une intégrale de Choquet engendrée aléatoirement, chaque réponse étant ensuite insérée dans $\mathcal{P}$. À chaque itération, nous calculons à la fois la valeur mMR et le *real regret*, ce dernier correspondant à la perte d'utilité réelle engendrée par la recommandation de l'alternative optimale pour le critère de décision Minimax Regret au lieu de l'alternative que le décideur préfère réellement ; cette différence d'utilité est tout simplement calculée avec l'intégrale de Choquet utilisée pour répondre aux différentes questions. Ces regrets sont exprimées sur une échelle normalisée, associant la valeur 1 à la quantité mMR initiale (c'est-à-dire, la valeur mMR obtenue avant d'avoir posé des questions). Rappelons que si la valeur mMR est égale à 0 à un moment donné, alors cela signifie que la solution optimale pour le Minimax Regret est l'alternative que le décideur préfère. Les vecteurs de performances des alternatives sont engendrés en sélectionnant aléatoirement des solutions Pareto-optimales d'un problème de sac à dos multi-objectifs (voir les expériences de la Section 2.1), de sorte à obtenir des vecteurs qui sont incomparables avec la dominance de Pareto (nécessitant donc des informations supplémentaires sur les préférences du décideur). La figure 3.1 montre les résultats que nous avons obtenus en moyennant sur 100 expériences.

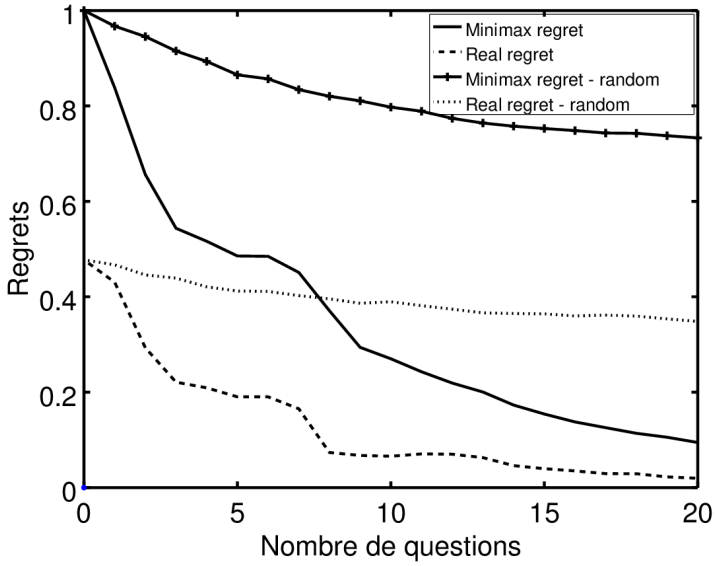


FIGURE 2.3 – Comparaison avec une génération de questions aléatoire ($q = 10$, 1000 alternatives).

Tout d'abord, nous voyons que la valeur mMR diminue bien plus rapidement avec notre stratégie de génération de questions qu'avec la stratégie de génération aléatoire. Par exemple, après environ 20 questions en moyenne, la valeur mMR est inférieure à 10% du regret initial avec notre stratégie, alors que cette valeur est toujours au dessus de 70% avec la stratégie aléatoire. Le même phénomène se produit avec les regrets réels. Par ailleurs, nous observons que le *real regret* est bien plus faible que la valeur mMR

à chaque itération (ce dernier fait a déjà été observé dans d'autres travaux, e.g., [Wang and Boutilier, 2003, Viappiani and Boutilier, 2009b, Boutilier et al., 2006]). Par exemple, avec notre stratégie, le *real regret* est inférieur à 10% du regret initial après environ 8 questions en moyenne alors que le mMR est au dessus de 30% de cette valeur à ce moment là.

Les expériences suivantes ont pour objectif d'évaluer les performances de notre stratégie d'élicitation en comparaison avec la stratégie CSS présentée en Section 1.4.2. Comme cette stratégie demande au décideur de comparer des alternatives réelles (et non des alternatives binaires à des profils constants), alors les PMR sont calculées en utilisant un solveur pour résoudre le programme linéaire (2.8)-(2.11). Par ailleurs, puisque les variables et les contraintes de ce programme sont en nombre exponentiel par rapport au nombre de critères, nous considérons ici des instances plus petites que précédemment ($q = 5$ et 150 alternatives). Comme dans toutes nos expériences, nous faisons appel au solveur Gurobi depuis un programme écrit en Java. Les résultats obtenus en moyennant 100 tests sont donnés en figure 2.4.

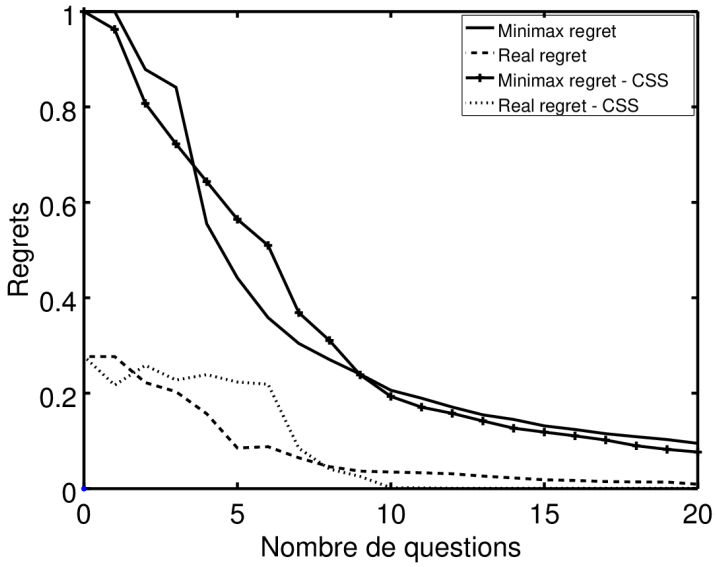


FIGURE 2.4 – Comparaison avec la stratégie CSS ($q = 5$, 150 alternatives).

Sur cette figure, nous pouvons voir que notre stratégie de génération de questions réalise des performances relativement similaires à la stratégie CSS en terme de réduction de regrets ; par exemple, après environ 20 questions en moyenne, les deux procédures parviennent à formuler une recommandation avec un regret dans le pire cas en dessous de 10% du regret initial. Toutefois, notre stratégie de génération de questions est plus avantageuse que la stratégie CSS en terme de temps. Pour nous en rendre compte, nous comparons maintenant les temps de calcul de mMR observés avec les méthodes suivantes :

- $\mathbf{M}_1$: les questions sont engendrées selon la stratégie CSS. Les PMR sont alors calculés en utilisant un solveur pour résoudre le programme linéaire (2.8)-(2.11).
- $\mathbf{M}_2$: les questions sont engendrées selon notre stratégie d'élicitation fondée sur les alternatives

binaires et les profils constants. Les PMR sont calculés en utilisant un solveur pour résoudre le programme linéaire simplifié (2.19)-(2.25).

- $\mathbf{M_3}$: les questions sont ici aussi engendrées selon notre stratégie d'élicitation fondée sur les alternatives binaires et profils constants, mais les PMR sont calculés en utilisant l'algorithme itératif présenté en Section 2.2.2 (au lieu d'avoir recours à un solveur).

Pour évaluer l'impact des données collectées sur les temps de résolution, nous reportons ci-après les résultats obtenus après respectivement 0, 10 et 20 itérations des procédures d'élicitation ; les expériences ont été réalisées sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire.

TABLE 2.2 – Comparaison des temps de calcul de mMR (en secondes).

q	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{P}\rvert$	M_1	M_2	M_3
5	150	0	4.473	3.846	0.005
5	150	10	4.833	2.594	0.001
5	150	20	4.805	2.820	0.001
5	1000	0	35.944	32.161	0.007
5	1000	10	25.471	24.129	0.004
5	1000	20	26.792	24.131	0.003
10	150	0	26.357	3.864	0.002
10	150	10	9.504	2.586	0.001
10	150	20	12.901	2.087	0.001
10	1000	0	276.067	57.795	0.014
10	1000	10	140.062	38.721	0.008
10	1000	20	218.528	34.499	0.007

Dans la table 2.2, nous observons que les temps de résolution avec M_1 augmentent drastiquement avec le nombre d'alternatives, notamment parce que celui-ci a un impact direct sur le nombre de PMR à calculer. Le même phénomène se produit en augmentant le nombre de critères du problème. Ceci s'explique par le fait que la formulation (2.8)-(2.11) contient un nombre de variables et de contraintes qui est exponentiel en le nombre de critères. Sans surprise, nous observons par ailleurs que la méthode M_2 est globalement plus rapide que la méthode M_1. En particulier, les performances de M_2 se dégradent moins rapidement que celles de M_1 lorsque le nombre de critères augmente ; ceci est dû au fait que le programme linéaire (2.19)-(2.25) fait intervenir uniquement un nombre linéaire de variables et contraintes. La stratégie M_3 est quant à elle beaucoup plus rapide que les deux autres (au moins de cinq ordres de grandeur). De plus, ses temps de calcul sont très peu impactés par le nombre de critères du problème et par le nombre d'alternatives à comparer. Pour donner une idée du gain de temps total, sur les instances avec 1000 alternatives et 10 critères, la stratégie M_3 engendre 50 questions en quelques minutes, alors que M_1 requiert plus de trois heures de calcul. En conclusion, en restreignant le type de questions, nous avons été capable de produire une procédure d'élicitation incrémentale très efficace pour les intégrales de Choquet, que ce soit en nombre de questions posées au décideur ou en temps de calcul.

2.3 Approche incrémentale pour le rangement et le tri avec l'intégrale de Choquet

Dans cette section, nous nous attelons à étendre l'approche d'élicitation incrémentale fondée sur le concept de regret, traditionnellement utilisée pour la problématique du choix, aux problèmes de rangement et de tri en décision multicritère.

2.3.1 Une méthode par choix itérés pour le rangement

Le rangement est une problématique de recherche ayant reçu une attention relativement importante de la part de diverses communautés, comme par exemple en psychologie (e.g., [Critchlow et al., 1991]), en apprentissage automatique (e.g., [Cohen et al., 1999, Herbrich et al., 2000, Hüllermeier et al., 2008, Ailon, 2012]) ou encore en gestion et en aide à la décision (e.g., [Fishburn, 1970, Keeney and Raiffa, 1976, Saaty, 1980, Jacquet-Lagrèze and Siskos, 1982, Brans and Vincke, 1985, Roy, 1996, Greco et al., 2008]). Cette problématique est bien plus complexe que celle du choix, car il s'agit de comparer toutes les alternatives entre elles pour produire un rangement.

Afin de déterminer le rangement reflétant les préférences du décideur du décideur, nous pouvons définir une notion de regret de type PMR entre deux rangements, pour ensuite appliquer l'approche d'élicitation incrémentale fondée sur les regrets. Bien que cette adaptation soit en théorie possible, celle-ci ne semble pas envisageable en pratique car le nombre de rangements possibles est exponentiel en le nombre d'alternatives du problème ; ainsi, le calcul systématique des PMR entre tous les rangements possibles paraît difficile à mettre en œuvre. De ce fait, nous abordons ici la problématique de rangement comme un problème de choix itérés, supposant que le décideur souhaite tout d'abord connaître la meilleure alternative, puis la seconde, et ainsi de suite, ce qui produit à la fin un rangement sur toutes les alternatives du problème. Ce processus décisionnel relativement naturel possède l'avantage de pouvoir être interrompu à une itération k si seules les k premières alternatives sont désirées. Lorsque la problématique de rangement est étudiée sous l'angle de choix itérés, celle-ci se présente comme une extension simple des problèmes de choix, et nous allons voir que le meilleur rangement peut être construit en exploitant les outils de résolution développés pour ces derniers problèmes.

Nous supposons ici que les préférences du décideur sont représentables par une intégrale de Choquet (cf. définition 16) dont la capacité v est imprécisément connue par le système. Dans ce contexte, une alternative x est nécessairement préférée à une alternative y si et seulement si l'inégalité $\mathrm{Ch}(x, v) \geq \mathrm{Ch}(y, v)$ est vraie pour toutes les capacités admissibles $v \in \Omega_{\mathcal{P}}$, i.e. si et seulement si $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}}) \leq 0$. Par ailleurs, si $\mathrm{MR}(x, \mathcal{X}, \Omega_{\mathcal{P}}) \leq 0$ pour une alternative $x \in \mathcal{X}$, alors x est nécessairement préférée à toutes les autres alternatives ; dans ce cas, l'alternative x peut être mise en première position du rangement sans craindre de faire une erreur. En revanche, si aucune alternative ne vérifie cette propriété, alors nous pouvons envisager de poser des questions au décideur jusqu'à obtenir $\mathrm{mMR}(\mathcal{X}, \Omega_{\mathcal{P}}) \leq 0$, ce qui nous permet de détecter une alternative nécessairement préférée à toutes les autres ; pour ce faire, nous

pouvons par exemple utiliser la stratégie de génération de questions que nous avons conçue spécialement pour l'intégrale de Choquet (cf. Section 2.2.3). Cette alternative peut ensuite être retirée de l'ensemble $\mathcal{X}$ des alternatives pour être placée en tête du rangement, et le processus de sélection peut être itéré sur le reste des alternatives. Pour déterminer l'alternative suivante, il convient bien évidemment de commencer l'élicitation avec l'ensemble de toutes les données collectées jusqu'à la fin de la première étape. L'alternative qui sera sélectionnée durant la deuxième étape sera alors mise en seconde position du rangement final, et ainsi de suite. Dans l'optique de réduire le nombre de questions posées, nous pouvons nous accorder un seuil de tolérance $\delta \geq 0$ et décider de sélectionner une alternative x si et seulement si l'inégalité $\mathrm{MR}(x, \mathcal{X}, \Omega_{\mathcal{P}}) \leq \delta$ est vérifiée. Cette méthode interactive par choix itérés est résumée dans l'algorithme 2.

Algorithm 2: Rangement par choix itérés.

Input: $\mathcal{X}$: un ensemble d'alternatives
Output: L : une liste représentant le rangement final

```
L ← ()
Z ← X
while Z ≠ ∅ do
    while mMR(Z, Ω_P) > δ do
        Poser une question au décideur et ajouter la réponse dans P
        Mettre à jour Ω_P en fonction de la réponse
    end
    Choisir z* dans arg min_{z∈Z} MR(z, Z, Ω_P)
    Ajouter z* à la fin de la liste L
    Retirer z* de l'ensemble Z
end
return L
```

Cet algorithme de rangement incrémental fondé sur le concept de regret satisfait la propriété suivante :

Proposition 10. *Pour toute paire d'alternatives x, y telles que x est placée devant y dans le rangement final, nous avons* $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}}) \leq \delta$*, où $\mathcal{P}$ est l'ensemble de toutes les données collectées lors de la construction du rangement.*

Démonstration. Soit $\mathcal{P}'$ l'ensemble des données collectées avant l'ajout de x dans le rangement. Puisque y est positionnée derrière x dans le rangement final, alors nous avons forcément $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}'}) \leq \delta$. De plus, comme $\mathcal{P}' \subseteq \mathcal{P}$, alors nous avons $\Omega_{\mathcal{P}'} \supseteq \Omega_{\mathcal{P}}$, ce qui nous permet de dériver les inégalités suivantes : $\mathrm{PMR}(x, y, \Omega_{\mathcal{P}}) \leq \mathrm{PMR}(x, y, \Omega_{\mathcal{P}'}) \leq \delta$. □

Cette proposition met en lumière une propriété particulièrement intéressante de notre procédure de rangement incrémental : aucun chaînage d'erreurs ne se produit en combinant les préférences locales déduites durant la construction du rangement.

Résultats expérimentaux

Nous présentons à présent quelques résultats expérimentaux sur notre procédure de rangement incrémental combinée à la procédure d'élicitation présentée en Section 2.2.3. Nous souhaitons étudier l'évolution du nombre de questions engendrées en fonction du nombre d'alternatives déjà insérées dans le rangement. Pour ce faire, nous réalisons des tests avec différents seuils de tolérance δ (0.05, 0.1 et 0.15) afin d'estimer son impact sur le nombre de questions posées durant l'élicitation. La figure 2.5 résume les résultats que nous avons obtenus en moyennant sur 50 tests, pour des instances à 1000 alternatives et 5 critères (les vecteurs de performances ont été engendrés comme dans la Section 2.1).

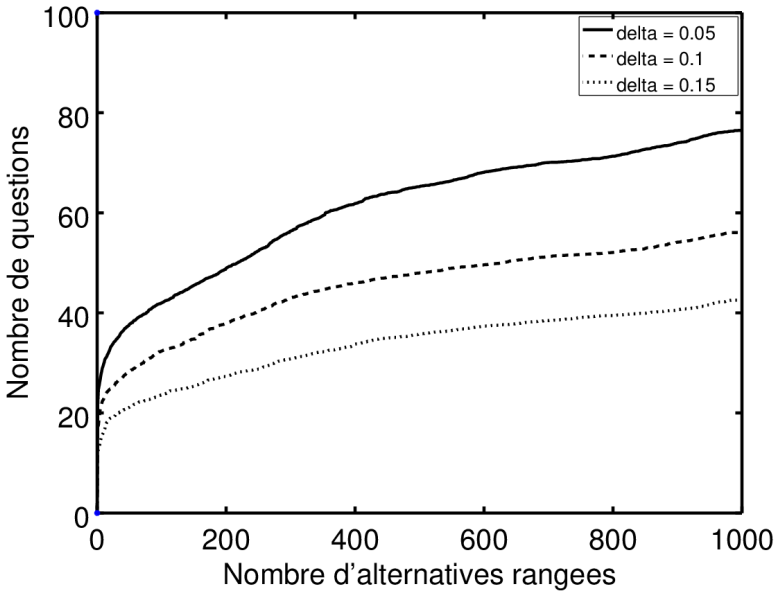


FIGURE 2.5 – Performance (en nombre questions) de la procédure de rangement incrémental par choix itérés ($q = 5$, 1000 alternatives).

Sans grande surprise, nous voyons que le nombre de questions engendrées réduit lorsque le seuil de tolérance δ augmente (puisque les exigences sur la garantie de performance s'affaiblissent). Par exemple, le nombre de questions nécessaires à la construction du rangement complet est divisé par deux quand le seuil de tolérance passe de 5% à 15% du regret initial. Par ailleurs, compléter le rangement après avoir déterminé la première alternative semble avoir un faible coût marginal : il nous faut moins de deux fois le nombre de questions posées durant la première étape. En réalité, nous avons observé empiriquement que le nombre d'alternatives a un impact relativement faible sur le nombre de questions posées au total, le nombre de questions engendrées à une étape du processus incrémental diminuant avec les itérations.

2.3.2 Méthodes pour le tri avec seuils de préférence

Dans le domaine de l'aide à la décision, les problèmes de choix (sélection de la meilleure alternative), de rangement (classement des alternatives de la meilleure à la plus mauvaise) et de tri (affectation des alternatives dans des catégories prédéfinies) constituent les trois grandes problématiques de références. Pour cette dernière problématique, il s'agit d'affecter toutes les alternatives du problème à des catégories prédéfinies, en se basant non seulement sur la qualité intrinsèque des alternatives, mais aussi sur les préférences subjectives du décideur. Par exemple, cette problématique apparaît naturellement lorsque le décideur souhaite décerner des distinctions spécifiques à des individus ou encore évaluer différentes demandes de crédits financiers. Dans la littérature, on peut distinguer l'approche quantitative *agréger puis comparer* de l'approche qualitative *comparer puis agréger* (e.g., [Perny, 2000, Grabisch and Perny, 2003]). Dans cette section, nous nous intéressons uniquement à cette première approche.

Lorsque les alternatives sont évaluées par le biais d'une fonction d'agrégation f_ω (e.g., somme pondérée, intégrale de Choquet), il semble naturel de trier les alternatives x par comparaison de leur valeur agrégée (ou utilité) $f_\omega(x)$ avec des seuils de préférence servant à délimiter les catégories. En procédant de cette manière, la problématique de tri est en quelque sorte vue comme une discrétisation de l'utilité en catégories résumant la qualité globale des alternatives. Cette approche standard en décision multicritère est aussi présente en apprentissage automatique, par exemple lorsque les algorithmes calculent une valeur numérique par alternative et affectent chaque alternative à la catégorie 0 ou 1 par comparaison de cette valeur avec un seuil de référence ; quand les catégories sont ordonnées, la communauté d'apprentissage automatique parle de *multipartite ranking* (e.g., [Fürnkranz et al., 2009, Quevedo et al., 2010, Uematsu and Lee, 2015]) ou encore de *instance ranking* (e.g., [Fürnkranz and Hüllermeier, 2011]). L'approche *agréger puis comparer* a été adoptée par de nombreuses organisations pour résoudre des problèmes de tri. Par exemple, le système de mentions du baccalauréat distingue les étudiants en comparant leur moyenne avec des notes prédéfinies.

Pour pouvoir mettre en œuvre cette approche, nous pouvons collecter des informations sur les préférences du décideur pour ensuite déterminer l'instance des paramètres ω permettant de modéliser au mieux ses préférences (e.g., [Devaud et al., 1980, Jacquet-Lagrèze, 1995, Zopounidis and Doumpos, 1999, 2002]) ou encore identifier des affectations nécessaires ou possibles (e.g. [Greco et al., 2010b]). Néanmoins, dans l'optique de trier toutes les alternatives en sollicitant le moins possible le décideur, nous adoptons ici une approche incrémentale fondée sur le concept de regret, permettant de réduire progressivement l'incertitude sur les paramètres du modèle jusqu'à être en mesure de déterminer le tri reflétant au mieux ses attentes. Pour présenter cette nouvelle approche, nous allons commencer par introduire des regrets adaptés aux problèmes de tri avec seuils de préférence. Puis, nous nous intéresserons à l'optimisation de ces nouveaux regrets dans le cas général, avant de nous concentrer sur les difficultés rencontrées lorsque f_ω est une intégrale de Choquet. Pour ce dernier modèle décisionnel, nous montrerons que les regrets peuvent être calculés en temps polynomial à condition de restreindre les interactions à un certain type de questions. Nous terminerons cette section en proposant et en évaluant une stratégie de génération de questions, spécialement conçue pour la problématique de tri avec seuils de préférence.

i) Notions de regret pour le tri avec seuils de préférence

Considérons un problème de tri multicritère où l'échelle d'utilité est divisée en k intervalles $[\alpha_\ell, \alpha_{\ell-1}]$, avec $\alpha_0 \geq \ldots \geq \alpha_k$, représentant les différentes catégories possibles pour notre problème de tri. Chaque alternative $x \in \mathcal{X}$ doit être affectée à la catégorie K_ℓ telle que $\alpha_\ell \leq f_\omega(x) \leq \alpha_{\ell-1}$, où $f_\omega(x)$ représente l'utilité de l'alternative x aux yeux du décideur ; dans le cas où l'égalité $f_\omega(x) = \alpha_\ell$ est vérifiée, nous supposons que le décideur est indifférent entre voir x dans la catégorie K_ℓ ou dans la catégorie $K_{\ell+1}$ (autrement dit, les deux affections sont valides). Supposons un instant que la procédure de tri se trompe en décidant d'affecter une alternative x à une certaine catégorie K_ℓ. Le regret (ou la perte) associé à cette erreur est d'autant plus important que la valeur $f_\omega(x)$ est éloignée des extrémités de l'intervalle définissant la catégorie K_ℓ. Plus précisément, ce regret est égal à $f_\omega(x) - \alpha_{\ell-1}$ si $f_\omega(x) \geq \alpha_{\ell-1}$ et vaut $\alpha_\ell - f_\omega(x)$ dans le cas contraire. En résumé, le regret associé à l'affectation d'une alternative $x \in \mathcal{X}$ à une catégorie $K_\ell, \ell \in \{1, \ldots, k\}$, est défini comme suit :

$$\mathrm{R}(x, K_\ell, \omega) = \max\Big\{ f_\omega(x) - \alpha_{\ell-1},\ \alpha_\ell - f_\omega(x),\ 0 \Big\} \tag{2.38}$$

Notons que cette définition de regret est cohérente avec le cas d'une affectation correcte ; en effet, il est facile de vérifier que ce regret est nul si $f_\omega(x)$ appartient à l'intervalle $[\alpha_\ell, \alpha_{\ell-1}]$. Par ailleurs, nous avons bien $\mathrm{R}(x, K_\ell, \omega) = 0$ et $\mathrm{R}(x, K_{\ell+1}, \omega) = 0$ lorsque $f_\omega(x) = \alpha_\ell$ (ces affectations sont donc bien valides).

Supposons à présent que les paramètres du modèle décisionnel sont connus de manière imprécise. Notons $\mathcal{P}$ l'ensemble des informations disponibles sur les préférences du décideur pour ce problème de tri. Par exemple, l'ensemble $\mathcal{P}$ peut contenir des informations du type "x est meilleure que y" ou encore "x appartient à la catégorie K_ℓ". Notons par ailleurs $\Omega_\mathcal{P}$ l'ensemble des instances de paramètres compatibles avec les informations contenues dans l'ensemble $\mathcal{P}$. Sans interaction possible avec le décideur, il s'agit de trier les alternatives dans les différentes catégories en tenant compte de l'imprécision sur les paramètres du modèle (résumée par l'ensemble $\Omega_\mathcal{P}$). Pour se prémunir du pire scénario possible, il convient de considérer le concept regret suivant :

Définition 38 (Max Regret (MR))**.** *Le MR d'une alternative $x \in \mathcal{X}$ par rapport à une catégorie K_ℓ est :*

$$\begin{aligned}\mathrm{MR}(x, K_\ell, \Omega_\mathcal{P}) &= \max_{\omega \in \Omega_\mathcal{P}} \mathrm{R}(x, K_\ell, \omega) \\ &= \max_{\omega \in \Omega_\mathcal{P}} \max\Big\{ f_\omega(x) - \alpha_{\ell-1},\ \alpha_\ell - f_\omega(x),\ 0 \Big\}.\end{aligned}$$

La quantité $\mathrm{MR}(x, K_\ell, \Omega_\mathcal{P})$ représente, par définition, le plus grand écart possible entre la valeur $f_\omega(x)$ et l'intervalle $[\alpha_\ell, \alpha_{\ell-1}]$ définissant la catégorie K_ℓ. Cette notion de regret nous permet d'introduire le concept suivant :

Définition 39 (Minimax Regret (mMR))**.** *Le mMR d'une alternative $x \in \mathcal{X}$ est défini par :*

$$\mathrm{mMR}(x, \Omega_\mathcal{P}) = \min_{\ell \in \{1,\ldots,k\}} \mathrm{MR}(x, K_\ell, \Omega_\mathcal{P})$$

Dans l'optique de minimiser les regrets, chaque alternative $x \in \mathcal{X}$ doit être affectée à la catégorie minimisant la valeur $\mathrm{MR}(x, K_\ell, \Omega_\mathcal{P})$; celle-ci est aussi appelée *catégorie optimale* au sens du mMR. La quantité $\mathrm{mMR}(x, \Omega_\mathcal{P})$ représente alors le plus grand écart d'utilité entre la valeur $f_\omega(x)$ et les extrémités de la catégorie optimale pour x.

Contrairement aux problèmes de choix, chaque alternative est ici associée à une valeur mMR et donc nous avons un vecteur de $|\mathcal{X}|$ valeurs mMR à gérer. Pour les problèmes de tri, nous avons donc besoin de définir une valeur agrégée des mMR pour évaluer la qualité de toute l'affection. Par exemple, nous pourrions utiliser une somme pondérée pour autoriser des compensations entre les différentes alternatives. Toutefois, nous faisons ici le choix de travailler avec la notion de regret suivante :

Définition 40 (Maximum Minimax Regret (MmMR))**.**

$$\mathrm{MmMR}(\mathcal{X}, \Omega_\mathcal{P}) = \max_{x \in \mathcal{X}} \mathrm{mMR}(x, \Omega_\mathcal{P})$$

Les valeurs mMR sont donc agrégées par l'opérateur maximum, ce choix paraissant être le plus cohérent avec la vision pessimiste du MR et offrant une garantie de performance par rapport au pire scénario possible. La valeur MmMR joue le rôle de mesure de la qualité de la décision (comme la valeur mMR dans les problèmes de choix) : plus la valeur MmMR est petite, plus l'erreur dans le pire scénario est faible. En particulier, lorsque $\mathrm{MmMR}(\mathcal{X}, \Omega_\mathcal{P}) = 0$, le tri obtenu en suivant les valeurs mMR est exactement celui que le décideur juge adapté.

Dans les situations où le décideur estime que la valeur MmMR est trop grande, nous pouvons envisager de lui poser des questions pour améliorer la qualité du tri. Pour limiter le nombre de questions, il s'agit de concevoir des procédures d'élicitation incrémentale, posant progressivement des questions au décideur, jusqu'à ce que la valeur MmMR devienne plus petite qu'un certain seuil de tolérance $\delta \geq 0$. Ceci est en effet réalisable car nous avons $\Omega_{\mathcal{P}'} \subseteq \Omega_\mathcal{P}$ pour tout $\mathcal{P}' \supseteq \mathcal{P}$, ce qui implique :

$$\begin{aligned}
&\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}'}) \leq \mathrm{MR}(x, K_\ell, \Omega_\mathcal{P}),\ \forall x \in \mathcal{X}, \forall \ell \in \{1, \ldots, k\} \\
&\mathrm{mMR}(x, \Omega_{\mathcal{P}'}) \leq \mathrm{mMR}(x, \Omega_\mathcal{P}),\ \forall x \in \mathcal{X} \\
&\mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}'}) \leq \mathrm{MmMR}(\mathcal{X}, \Omega_\mathcal{P})
\end{aligned}$$

Ces inégalités nous informent que la valeur MmMR ne peut augmenter en ajoutant de nouvelles données dans l'ensemble $\mathcal{P}$; dans nos expériences, nous verrons que cette valeur décroît strictement en pratique (à condition de bien choisir les questions). Pour évaluer la pertinence d'une question, nous procédons comme dans les problèmes de choix en utilisant la notion de valeur de l'information "myope" suivante :

Définition 41 (Worst-case maximum minimax regret (WMmMR))**.** *Le WMmMR d'une question d est :*

$$\mathrm{WMmMR}(d, \Omega_\mathcal{P}) = \max_{r \in \mathcal{R}_d} \mathrm{MmMR}(\Omega_{\mathcal{P} \cup \{r\}})$$

où $\mathcal{R}_d$ est l'ensemble des réponses possibles à la question d.

D'après le critère de décision WMmMR, la prochaine question à poser doit être sélectionnée parmi les éléments de l'ensemble $\arg\min_{d\in D} \mathrm{WMmMR}(d, \Omega_{\mathcal{P}})$, où D est l'ensemble de toutes les questions possibles. Par définition, poser une question optimale pour le critère WMmMR permet de garantir la plus grande réduction de la valeur MmMR dans le pire scénario de réponse. Cependant, déterminer une question optimale pour ce critère est une tâche bien trop coûteuse en pratique car l'ensemble D des questions possibles est généralement très grand. Dans ces situations, il est préférable d'utiliser une heuristique efficace de ce critère pour que la durée totale du questionnaire ne soit pas trop longue. Par exemple, nous pouvons décider de poser uniquement des questions de comparaison au décideur, en choisissant avec soin les deux alternatives à comparer. Rappelons que pour les problèmes de choix, une heuristique efficace du WmMR a déjà été proposée, appelée CSS, consistant à demander au décideur de comparer les deux alternatives induisant la valeur courante du mMR (cf. Section 1.4.2). Néanmoins, dans le cadre du tri, la valeur MmMR courante ne suggère pas directement deux alternatives qu'il convient de comparer pour réduire le plus possible la valeur MmMR. À la place, nous connaissons la ou les alternatives dont l'affectation explique la valeur MmMR courante. Par conséquent, dans la même logique que la stratégie CSS, nous pouvons envisager de demander au décideur, à chaque itération, de positionner l'alternative maximisant la valeur mMR dans la catégorie qui lui semble la plus appropriée. De cette façon, nous centrons bien le questionnaire sur les alternatives induisant la valeur MmMR.

ii) Détermination du tri optimal au sens du critère MmMR

Pour trier les alternatives selon le critère MmMR, nous pouvons tout d'abord calculer la quantité $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}})$ pour tout $x \in \mathcal{X}$ et tout $\ell \in \{1, \ldots, k\}$, puis affecter chaque alternative $x \in \mathcal{X}$ à la catégorie $K_\ell, \ell \in \{1, \ldots, k\}$, minimisant la quantité $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}})$. Le calcul des valeurs $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}})$ peut être réalisé de manière efficace en exploitant le résultat suivant :

Proposition 11. *Pour toute alternative $x \in \mathcal{X}$ et toute catégorie $K_\ell, \ell \in \{1, \ldots, k\}$, nous avons :*

$$\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) = \max\Big\{ f_x^{\top} - \alpha_{\ell-1},\ \alpha_\ell - f_x^{\perp},\ 0 \Big\}$$

où $f_x^{\top} = \max\limits_{\omega\in\Omega_{\mathcal{P}}} f_\omega(x)$ et $f_x^{\perp} = \min\limits_{\omega\in\Omega_{\mathcal{P}}} f_\omega(x)$.

Démonstration. Pour toute alternative $x \in \mathcal{X}$:

$$\begin{aligned}
\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) &= \max_{\omega\in\Omega_{\mathcal{P}}} \mathrm{R}(x, K_\ell, \omega) \\
&= \max_{\omega\in\Omega_{\mathcal{P}}} \max\Big\{ f_\omega(x) - \alpha_{\ell-1},\ \alpha_\ell - f_\omega(x),\ 0 \Big\} \\
&= \max\Big\{ \max_{\omega\in\Omega_{\mathcal{P}}} \{f_\omega(x) - \alpha_{\ell-1}\}, \max_{\omega\in\Omega_{\mathcal{P}}} \{\alpha_\ell - f_\omega(x)\},\ \max_{\omega\in\Omega_{\mathcal{P}}} \{0\} \Big\} \\
&= \max\Big\{ \max_{\omega\in\Omega_{\mathcal{P}}} \{f_\omega(x)\} - \alpha_{\ell-1},\ \alpha_\ell - \min_{\omega\in\Omega_{\mathcal{P}}} \{f_\omega(x)\},\ 0 \Big\} \\
&= \max\Big\{ f_x^{\top} - \alpha_{\ell-1},\ \alpha_\ell - f_x^{\perp},\ 0 \Big\}
\end{aligned}$$

□

Ainsi, pour déterminer la catégorie optimale de chaque alternative, il suffit de déterminer les valeurs $\max_{\omega\in\Omega_{\mathcal{P}}} f_\omega(x)$ et $\min_{\omega\in\Omega_{\mathcal{P}}} f_\omega(x)$, puis de calculer $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}})$ pour chaque catégorie K_ℓ, $\ell \in \{1,\ldots,k\}$, en utilisant la formule de la proposition 11. En réalité, les calculs peuvent être davantage simplifiés en utilisant la proposition suivante :

Proposition 12. *Pour toute alternative $x \in \mathcal{X}$, la catégorie K_ℓ vérifiant*

$$\alpha_\ell \leq \frac{f_x^\top + f_x^\bot}{2} \leq \alpha_{\ell-1}$$

est la catégorie optimale pour x au sens du mMR*, où $f_x^\top = \max\limits_{\omega\in\Omega_{\mathcal{P}}} f_\omega(x)$ et $f_x^\bot = \min\limits_{\omega\in\Omega_{\mathcal{P}}} f_\omega(x)$.*

Démonstration. Il s'agit de montrer que l'inégalité $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) \leq \mathrm{MR}(x, K_{\ell'}, \Omega_{\mathcal{P}})$ est vérifiée pour tout $\ell' \in \{1,\ldots,k\}\backslash\{\ell\}$. Pour ce faire, nous allons tout d'abord montrer que l'inégalité $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) \leq (f_x^\top - f_x^\bot)/2$ est vraie, puis nous allons prouver que l'inégalité $\mathrm{MR}(x, K_{\ell'}, \Omega_{\mathcal{P}}) \geq (f_x^\top - f_x^\bot)/2$ est vérifiée pour tout $\ell' \in \{1,\ldots,k\}\backslash\{\ell\}$.

Puisque l'inégalité $(f_x^\top + f_x^\bot)/2 \leq \alpha_{\ell-1}$ est vraie par définition de la catégorie K_ℓ, alors nous avons $f_x^\top - \alpha_{\ell-1} \leq f_x^\top - (f_x^\top + f_x^\bot)/2 = (f_x^\top - f_x^\bot)/2$. Par ailleurs, comme l'inégalité $\alpha_\ell \leq (f_x^\top + f_x^\bot)/2$ est aussi vraie par définition de la catégorie K_ℓ, alors nous avons $\alpha_\ell - f_x^\bot \leq (f_x^\top + f_x^\bot)/2 - f_x^\bot = (f_x^\top - f_x^\bot)/2$. Par conséquent, d'après la proposition 11, nous avons $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) = \max\{f_x^\top - \alpha_{\ell-1},\ \alpha_\ell - f_x^\bot,\ 0\} \leq (f_x^\top - f_x^\bot)/2$. Montrons à présent que l'inégalité $\mathrm{MR}(x, K_{\ell'}, \Omega_{\mathcal{P}}) \geq (f_x^\top - f_x^\bot)/2$ est vérifiée pour tout $\ell' \in \{1,\ldots,k\}\backslash\{\ell\}$. Pour tout $\ell' > \ell$, nous avons :

$$\begin{aligned}
\mathrm{MR}(x, K_{\ell'}, \Omega_{\mathcal{P}}) &\geq f_x^\top - \alpha_{\ell'-1} \quad \text{d'après la proposition 11} \\
&= f_x^\top - \alpha_\ell + \alpha_\ell - \alpha_{\ell'-1} \\
&\geq f_x^\top - (f_x^\top + f_x^\bot)/2 + \alpha_\ell - \alpha_{\ell'-1} \quad \text{par définition de } K_\ell \\
&= (f_x^\top - f_x^\bot)/2 + \alpha_\ell - \alpha_{\ell'-1} \\
&\geq (f_x^\top - f_x^\bot)/2 \quad \text{puisque } \ell \leq \ell' - 1
\end{aligned}$$

Pour tout $\ell' < \ell$, nous avons :

$$\begin{aligned}
\mathrm{MR}(x, K_{\ell'}, \Omega_{\mathcal{P}}) &\geq \alpha_{\ell'} - f_x^\bot \quad \text{d'après la proposition 11} \\
&= \alpha_{\ell'} - \alpha_{\ell-1} + \alpha_{\ell-1} - f_x^\bot \\
&\geq \alpha_{\ell'} - \alpha_{\ell-1} + (f_x^\top + f_x^\bot)/2 - f_x^\bot \quad \text{par définition de } K_\ell \\
&= \alpha_{\ell'} - \alpha_{\ell-1} + (f_x^\top - f_x^\bot)/2 \\
&\geq (f_x^\top - f_x^\bot)/2 \quad \text{car } \ell - 1 \geq \ell'
\end{aligned}$$

Par conséquent, nous avons bien $\mathrm{MR}(x, K_{\ell'}, \Omega_{\mathcal{P}}) \geq (f_x^\top - f_x^\bot)/2$ pour tout $\ell' \in \{1,\ldots,k\}\backslash\{\ell\}$. Comme par ailleurs nous avons $\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) \leq (f_x^\top - f_x^\bot)/2$, nous en déduisons que la catégorie K_ℓ est la catégorie optimale pour l'alternative x au sens du mMR. □

D'après la proposition 12, pour déterminer la catégorie optimale d'une alternative x, il suffit de calculer la quantité $(f_x^\top + f_x^\perp)/2$ puis d'identifier la catégorie dont l'intervalle représentatif contient cette valeur. De ce fait, la seule source de complexité subsistante réside dans la détermination des valeurs $f_x^\top$ et $f_x^\perp$ pour chaque alternative $x \in \mathcal{X}$.

Lorsque f_ω est une fonction linéaire en ω, toute donnée de type "je préfère x à y", notée (x, y), induit la contrainte linéaire $f_\omega(x) \geq f_\omega(y)$ sur l'espace des paramètres. Par ailleurs, toute donnée de type "x doit appartenir à la catégorie K_ℓ", notée (x, K_ℓ), impose deux contraintes linéaires sur l'espace des paramètres : $f_\omega(x) \geq \alpha_\ell$ et $f_\omega(x) \leq \alpha_{\ell-1}$. Par conséquent, l'ensemble $\Omega_\mathcal{P}$ est entièrement décrit par des contraintes linéaires. Ce dernier fait nous permet de formuler les problèmes d'optimisation $f_x^\top = \max_{\omega\in\Omega_\mathcal{P}} f_\omega(x)$ et $f_x^\perp = \min_{\omega\in\Omega_\mathcal{P}} f_\omega(x)$ comme des programmes linéaires. Pour des modèles simples comme la somme pondérée (cf. définition 11), le calcul des valeurs $f_x^\top$ et $f_x^\perp$ peut donc être réalisé efficacement par programmation linéaire. En revanche, pour l'intégrale de Choquet, qui est aussi un agrégateur linéaire en ses paramètres, nous allons voir que le calcul de ces valeurs n'est pas aussi simple.

iii) Optimisation du critère MmMR avec l'intégrale de Choquet

Dans cette sous-section, nous supposons que les préférences du décideur sont représentables par une intégrale de Choquet (cf. définition 16) ; autrement dit, la valeur $f_\omega(x)$ est ici égale à la quantité $\mathrm{Ch}(x, v) = \sum_{j=1}^q \left[x_{(j)} - x_{(j-1)}\right] v_{X_{(j)}}$, la capacité v jouant le rôle de ω. L'espace des paramètres $\Omega_\mathcal{P}$ correspond donc ici à l'ensemble des capacités normalisées (cf. définition 13) compatibles avec les préférences observées (contenues dans l'ensemble $\mathcal{P}$). Dans ce contexte, il s'agit pour nous de déterminer efficacement la catégorie optimale de chaque alternative $x \in \mathcal{X}$ au sens du critère MmMR (ce qui permet aussi de connaître la valeur MmMR représentant la garantie de performance).

Dans la sous-section précédente, nous avons vu que la catégorie optimale d'une alternative $x \in \mathcal{X}$ peut être identifiée facilement à partir des valeurs $f_x^\top$ et $f_x^\perp$ (cf. proposition 12). En notant v_A la variable représentant la valeur $v(A)$ pour tout $A \subseteq \mathcal{Q}$, calculer la valeur $f_x^\top$ avec l'intégrale de Choquet revient à résoudre le programme linéaire suivant :

$$\max_{v} \quad \sum_{j=1}^{q} \left[x_{(j)} - x_{(j-1)}\right] v_{X_{(j)}} \tag{2.39}$$

$$s.c. \quad v_\emptyset = 0,\ v_\mathcal{Q} = 1 \tag{2.40}$$

$$v_A \leq v_{A\cup\{j\}},\ \forall A \subset \mathcal{Q},\ \forall j \in \mathcal{Q}\backslash A \tag{2.41}$$

$$\sum_{j=1}^{q} \left[a_{(j)} - a_{(j-1)}\right] v_{A_{(j)}} \geq \sum_{j=1}^{q} \left[b_{(j)} - b_{(j-1)}\right] v_{B_{(j)}},\ \forall (a, b) \in \mathcal{P} \tag{2.42}$$

$$\alpha_\ell \leq \sum_{j=1}^{q} \left[a_{(j)} - a_{(j-1)}\right] v_{A_{(j)}} \leq \alpha_{\ell-1},\ \forall (a, K_\ell) \in \mathcal{P} \tag{2.43}$$

où les équations (2.40)-(2.41) permettent de garantir que v est une fonction normalisée et monotone

(c'est-à-dire une capacité) et les équations (2.42)-(2.43) assurent que v est bien compatible avec les préférences observées et stockées dans $\mathcal{P}$. Le calcul de $f_x^{\perp}$ peut se faire en résolvant exactement le même programme linéaire, à la différence près que la fonction objectif doit être minimisée ; dans la suite, ces programmes linéaires seront appelés $\mathrm{LP}_x^{\top}$ et $\mathrm{LP}_x^{\perp}$ respectivement. Bien que ces deux programmes contiennent un nombre de variables et de contraintes qui est exponentiel en le nombre de critères, nous pouvons dériver des formulations plus compactes en considérant le même type de questions que dans la Section 2.2, c'est-à-dire demandant au décideur de comparer des alternatives binaires de type $1A0$, $A \subseteq \mathcal{Q}$, avec des profils d'utilité constants de type $\Lambda = (\lambda, \ldots, \lambda)$. En effet, remarquons que la fonction objectif des programmes $\mathrm{LP}_x^{\top}$ et $\mathrm{LP}_x^{\perp}$ implique uniquement les ensembles de niveau de l'alternative x, autrement dit les sous-ensembles de critères de l'ensemble suivant (cf. définition 15) :

$$\mathcal{A}_x = \{X_{(j)} : j \in \mathcal{Q}\}$$

Par conséquent, comme pour les problèmes de choix, nous allons pouvoir reformuler ces programmes linéaires de sorte à obtenir un nombre de variables et de contraintes drastiquement plus faible. Plus précisément, nous avons vu tout d'abord que si le décideur compare uniquement des alternatives binaires à des profils constants, alors les contraintes de préférences des équations (2.42) et (2.43) peuvent être remplacées par des contraintes de borne de type "$l_A \leq v_A \leq u_A$" pour tous les sous-ensembles $A \subseteq \mathcal{Q}$, avec $l_A \leq l_B$ et $u_A \leq u_B$ pour tout $A \subset B \subseteq \mathcal{Q}$. Ensuite, en utilisant la proposition 4 avec $\mathcal{A} = \mathcal{A}_x$, nous déduisons que toutes les contraintes de monotonie impliquant des variables non présentes dans la fonction objectif (c'est-à-dire $v_A, A \notin \mathcal{A}_x$) peuvent être retirées des programmes linéaires $\mathrm{LP}_x^{\top}$ et $\mathrm{LP}_x^{\perp}$ puisque celles-ci ne jouent en réalité aucun rôle durant l'optimisation. En procédant de cette façon, nous obtenons des formulations linéaires avec seulement q variables (correspondant aux éléments de $\mathcal{A}_x$) et $q-1$ contraintes de monotonie. En particulier, le programme $\mathrm{LP}_x^{\top}$ se réécrit comme suit :

$$\max_{v} \quad \sum_{j=1}^{q} \left[x_{(j)} - x_{(j-1)}\right] v_{X_{(j)}} \tag{2.44}$$

$$\begin{aligned} s.c. \quad & v_{X_{(j+1)}} \leq v_{X_{(j)}}, \ \forall j \in \{1, \ldots, q-1\} \\ & l_{X_{(j)}} \leq v_{X_{(j)}} \leq u_{X_{(j)}}, \ \forall j \in \mathcal{Q} \end{aligned} \tag{2.45}$$

La nouvelle formulation associée au calcul de $\mathrm{LP}_x^{\perp}$ s'obtient simplement en remplaçant dans ce dernier programme l'opérateur maximum par l'opérateur minimum. Ces programmes linéaires plus compactes peuvent être résolus en temps polynomial en utilisant la programmation linéaire. Néanmoins, en exploitant la structure très particulière de ces derniers, nous pouvons déterminer leur valeur optimale encore plus efficacement, comme le montre la proposition suivante :

Proposition 13. *Pour toute alternative $x \in \mathcal{X}$:*

- *La solution $v_A = u_A$ pour tout $A \in \mathcal{A}_x$ est une solution optimale de la version simplifiée de $LP_x^{\top}$.*
- *La solution $v_A = l_A$ pour tout $A \in \mathcal{A}_x$ est une solution optimale de la version simplifiée de $LP_x^{\perp}$.*

Démonstration. Montrons que la solution $v_A = u_A$ pour tout $A \in \mathcal{A}_x$ est une solution optimale de la version simplifiée de $\mathrm{LP}_x^\top$ (le deuxième point de la proposition se montre avec des arguments similaires). Pour tout $A \in \mathcal{A}_x$, notons c_A le coefficient de la variable de décision v_A dans la fonction objectif (2.44). Avec cette notation, la fonction objectif (2.44) se réécrit $\sum_{A\in\mathcal{A}_x} c_A v_A$. Par définition des intégrales de Choquet, nous avons $c_A = x_{(j)} - x_{(j-1)} \geq 0$ pour tout $A \in \mathcal{A}_x = \{X_{(j)} : j \in \mathcal{Q}\}$. Puisque $c_A \geq 0$ et $v_A \leq u_A$ pour tout $A \in \mathcal{A}_x$, alors la quantité $\sum_{A\in\mathcal{A}_x} c_A u_A$ représente une borne supérieure de la valeur optimale du programme simplifié. Pour pouvoir conclure la preuve, nous devons maintenant montrer que l'instanciation $v_A = u_A$ est une solution réalisable du programme simplifié. Pour ce faire, il suffit de prouver que les contraintes de monotonie données en équation (2.45) sont vérifiées par cette instanciation. Ces contraintes sont satisfaites si et seulement si $u_{X_{(j+1)}} \leq u_{X_{(j)}}$ pour tout $j \in \{1, \ldots, q-1\}$. Puisque l'inégalité $u_A \leq u_B$ est vraie pour tous sous-ensembles $A \subseteq B \subseteq \mathcal{Q}$ par construction des intervalles $[l_A, u_A]$, $A \subseteq \mathcal{Q}$, alors nous avons bien $u_{X_{(j+1)}} \leq u_{X_{(j)}}$ pour tout $j \in \{1, \ldots, q-1\}$; ceci nous permet de conclure la preuve. □

La proposition 13 nous permet de résoudre les problèmes $f_x^\top$ et $f_x^\perp$ sans avoir recours à la programmation linéaire. Plus précisément, pour déterminer $f_x^\top$ (respectivement $f_x^\perp$), il suffit de considérer l'instanciation $v_A = u_A$ pour tout $A \subseteq \mathcal{Q}$ (respectivement $v_A = l_A$ pour tout $A \subseteq \mathcal{Q}$), et de calculer l'intégrale de Choquet de x par rapport à cette capacité (c'est-à-dire $\sum_{A\in\mathcal{A}_x} c_A v_A$).

Exemple 19. *Considérons un problème de tri défini sur 5 critères et 5 catégories délimitées par les seuils suivants :* $\alpha_0 = 1$, $\alpha_1 = 0.9$, $\alpha_2 = 0.7$, $\alpha_3 = 0.4$, $\alpha_4 = 0.2$ *et* $\alpha_5 = 0$. *L'objectif est de déterminer la catégorie optimale pour l'alternative* $x = (1, 0.8, 0.4, 0.5, 0.1)$ *au sens du critère* mMR. *Supposons que les intervalles* $[l_A, u_A]$, $A \in \{X_{(j)} : j \in \mathcal{Q}\}$, *sont les mêmes que ceux de l'exemple 18. Dans ce cas, le calcul de* $f_x^\top = \max_{v\in\Omega_\mathcal{P}} \mathrm{Ch}(x, v)$ *peut être effectué en résolvant le programme linéaire suivant (version simplifiée de* $LP_x^\top$*) :*

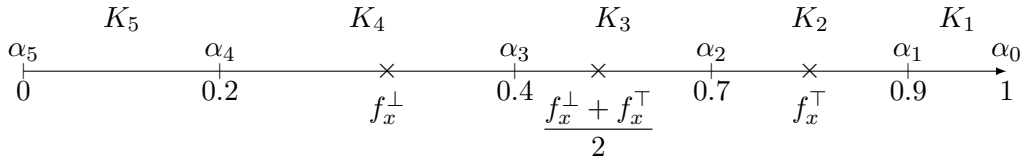
$$
\begin{aligned}
\max_{v} \quad & 0.1v_{\mathcal{Q}} + 0.3v_{\{1,2,3,4\}} + 0.1v_{\{1,2,4\}} + 0.3v_{\{1,2\}} + 0.2v_{\{1\}} \\
s.c. \quad & v_{\{1\}} \leq v_{\{1,2\}} \leq v_{\{1,2,4\}} \leq v_{\{1,2,3,4\}} \leq v_{\mathcal{Q}} \\
& 0 \leq v_{\{1\}} \leq 0.5,\ 0.2 \leq v_{\{1,2\}} \leq 0.7,\ 0.3 \leq v_{\{1,2,4\}} \leq 0.9,\ 0.6 \leq v_{\{1,2,3,4\}} \leq 1,\ 1 \leq v_{\mathcal{Q}} \leq 1
\end{aligned}
$$

La valeur $f_x^\perp = \min_{v\in\Omega_\mathcal{P}} \mathrm{Ch}(x, v)$ *peut quant à elle être calculée en résolvant le programme obtenu par remplacement de l'opérateur maximum par l'opérateur minimum. Ces optimisations peuvent en réalité être réalisées sans programmation linéaire, puisque les solutions optimales sont explicitées dans la proposition 13. Plus précisément, nous avons :*

$$
\begin{aligned}
f_x^\top &= 0.1u_{\mathcal{Q}} + 0.3u_{\{1,2,3,4\}} + 0.1u_{\{1,2,4\}} + 0.3u_{\{1,2\}} + 0.2u_{\{1\}} \\
&= 0.1 \times 1 + 0.3 \times 1 + 0.1 \times 0.9 + 0.3 \times 0.7 + 0.2 \times 0.5 \\
&= 0.8
\end{aligned}
$$

$$\begin{aligned} f_x^{\perp} &= 0.1 l_{\mathcal{Q}} + 0.3 l_{\{1,2,3,4\}} + 0.1 l_{\{1,2,4\}} + 0.3 l_{\{1,2\}} + 0.2\ l_{\{1\}} \\ &= 0.1 \times 1 + 0.3 \times 0.6 + 0.1 \times 0.3 + 0.3 \times 0.2 + 0.2 \times 0 \\ &= 0.37 \end{aligned}$$

Comme $(f_x^{\top} + f_x^{\perp})/2 = 0.585 \in [\alpha_3, \alpha_2]$*, alors nous pouvons conclure que* K_3 *est la catégorie optimale pour l'alternative* x *au sens du* MmMR *(d'après la proposition 12). La figure ci-dessous permet de résumer graphiquement ce problème d'affectation.*



iv) Une stratégie d'élicitation pour les intégrales de Choquet

Dans la sous-section précédente, nous avons vu que le tri optimal au sens du MmMR peut être identifié de manière efficace lorsque $\mathcal{P}$ contient uniquement des données de la forme $(1A0, \Lambda)$ ou $(\Lambda, 1A0)$. La valeur MmMR représente alors la garantie de performance associée à la recommandation de ce tri au décideur. Dans cette sous-section, nous nous intéressons à la conception d'une stratégie d'élicitation permettant de réduire efficacement la valeur MmMR à chaque itération par le biais de questions permettant de collecter des informations sous la forme souhaitée. Pour pouvoir introduire formellement notre stratégie, nous notons ici (A, λ) la question consistant à demander au décideur de comparer les alternatives $1A0$ et $\Lambda = (\lambda, \ldots, \lambda)$, avec $A \subset \mathcal{Q}$ et $\lambda \in [l_A, u_A]$.

Pour choisir la prochaine question à poser, nous utilisons le critère de sélection WMmMR (cf. définition 41). D'après ce critère, une question optimale est définie par une paire (A, λ) minimisant la valeur suivante :

$$\text{WMmMR}((A,\lambda), \Omega_{\mathcal{P}}) = \max\Big\{ \text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P} \cup \{(1A0, \Lambda)\}}), \text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P} \cup \{(\Lambda, 1A0)\}}) \Big\}$$

Par définition, un question optimale pour le critère WMmMR permet de minimiser la valeur MmMR dans le pire scénario de réponse du décideur. Pour pouvoir identifier une telle question, nous devons déterminer, pour tout $A \subseteq \mathcal{Q}$, la quantité suivante :

$$\begin{aligned} \lambda_A &= \arg \min_{\lambda \in [l_A, u_A]} \text{WMmMR}((A,\lambda), \Omega_{\mathcal{P}}) \\ &= \arg \min_{\lambda \in [l_A, u_A]} \max\Big\{ \text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P} \cup \{(1A0, \Lambda)\}}), \text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P} \cup \{(\Lambda, 1A0)\}}) \Big\} \end{aligned}$$

et sélectionner ensuite un ensemble A^* dans $\arg\min_{A \subseteq \mathcal{Q}} \text{WMmMR}((A, \lambda_A), \Omega_{\mathcal{P}})$; de cette façon, la paire $(A^*,\ \lambda_{A^*})$ constitue par construction une question optimale pour le critère WMmMR.

Pour tout ensemble $A \subseteq \mathcal{Q}$, la détermination de la valeur λ_A peut être réalisée relativement simplement en appliquant un algorithme dichotomique classique, en utilisant des arguments similaires à ceux avancés dans le cadre de la problématique de choix (cf. Section 2.2.3). En effet, les fonctions $\text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}\cup\{(1A0,\Lambda)\}})$ et $\text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}\cup\{(\Lambda,1A0)\}})$ sont respectivement décroissante et croissante, et atteignent la même valeur maximale (c'est-à-dire la valeur $\text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}})$).

Dans le cas où l'égalité $\text{WMmMR}((A^*, \lambda_{A^*}), \Omega_{\mathcal{P}}) = \text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}\cup\{(1A0,\Lambda)\}})$ est vraie, la question (A^*, λ_{A^*}) ne permet pas de garantir une réduction de regret dans le pire scénario de réponse. Dans ce cas, pour discriminer entre les différentes questions possibles, il convient de faire usage d'un critère de sélection moins conservateur que le critère WMmMR. Nous proposons par exemple de poser la question de type $(A, \frac{l_A+u_A}{2})$ qui minimise la moyenne des valeurs MmMR obtenues dans les différents scénarios de réponses.

À chaque itération de la procédure d'élicitation, pour déterminer la prochaine question, nous devons lancer un algorithme dichotomique sur tous les sous-ensembles $A \subset \mathcal{Q}$ non vides, qui sont au nombre de $2^q - 2$. Puisque ce nombre croît exponentiellement avec le nombre de critères du problème, il convient de proposer une heuristique de ce critère de sélection pour réduire les temps de calcul entre chaque question. Nous proposons ici de nous concentrer sur les sous-ensembles A de $\mathcal{Q}$ qui sont directement impliqués dans le calcul de $\text{mMR}(x, \Omega_{\mathcal{P}})$, où x est une alternative qui maximise la valeur mMR ; cette alternative est en partie "responsable" de la valeur MmMR courante. Plus précisément, nous nous intéressons uniquement aux q ensembles de niveau de x, c'est-à-dire aux éléments de l'ensemble $\mathcal{A}_x = \{X_{(j)} : j \in \mathcal{Q}\}$. De cette manière, notre heuristique produit uniquement des contraintes sur des variables qui expliquent la valeur $\text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}})$ courante, ce qui va potentiellement permettre de réduire cette dernière valeur.

v) Résultats expérimentaux

Les premiers tests réalisés dans cette sous-section ont pour objectif d'évaluer les performances de notre stratégie d'élicitation présentée en Section 2.3.2 (cette stratégie sera appelée S0 ci-après). Dans la Section 2.2.4, nous avons vu que le type de questions utilisé par la stratégie S0 permet de réduire considérablement les temps de résolution. C'est pourquoi, les expériences qui suivent visent uniquement à évaluer cette stratégie sur la base du nombre de questions engendrées. Rappelons que la stratégie S0 repose sur des questions impliquant une alternative binaire $1A0$ et un profil constant $\Lambda = (\lambda, \ldots, \lambda)$, choisis soigneusement à chaque itération. Par conséquent, comme base de comparaison, nous considérons la stratégie de génération de questions consistant à sélectionner, à chaque itération, l'ensemble $A \subseteq \mathcal{Q}$ et la valeur $\lambda \in [l_A, u_A]$ de manière aléatoire.

Dans ces expériences, les vecteurs de performances des alternatives sont engendrés uniformément dans $[0, 1]^q$ et les catégories sont définies en divisant uniformément l'échelle d'utilité. Par ailleurs, nous commençons avec un ensemble de données $\mathcal{P}$ vide, et nous simulons les réponses aux questions avec une intégrale de Choquet engendrée aléatoirement. À chaque itération, en plus de la valeur MmMR, nous calculons aussi le *maximum real regret*. Ce dernier regret représente la plus grande erreur commise en affectant une alternative à la catégorie optimale au sens du mMR au lieu de la positionner dans la

catégorie que le décideur considère être la mieux adaptée ; ce regret est tout simplement obtenu avec la formule (2.38) appliquée à l'intégrale de Choquet simulant les réponses. Comme dans tous nos tests, les regrets sont ici normalisés pour appartenir à l'intervalle unité afin de faciliter l'interprétation des diminutions de regrets. Les résultats obtenus en moyennant sur 100 tests sont présentés dans la figure 2.6.

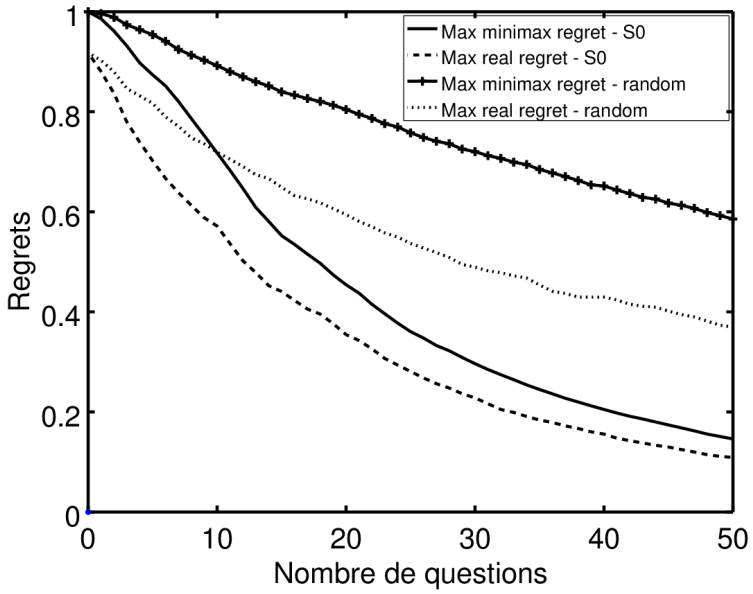


FIGURE 2.6 – Comparaison avec la stratégie aléatoire (5 critères, 1000 alternatives, 10 catégories).

La figure 2.6 montre que la valeur MmMR réduit beaucoup plus rapidement avec la stratégie S0 qu'avec la stratégie aléatoire. Nous observons la même chose pour le *maximum real regret*. Par exemple, après 30 questions en moyenne, le *maximum real regret* obtenu avec S0 est en dessous de 20% du regret initial alors que celui observé avec la stratégie aléatoire est toujours au dessus de 50% de cette valeur. Remarquons toutefois que le *maximum real regret* réduit moins rapidement que le *real regret* dans les problèmes de choix (cf. figure 3.1). Cependant, avec S0, il s'avère que la moyenne des *real regrets* sur les alternatives à classer est en dessous de 10% de la valeur initiale après seulement 5 questions en moyenne. Ce dernier fait montre que la plupart des alternatives sont correctement affectées relativement rapidement en pratique.

Rappelons que la stratégie S0 concentre ses efforts sur une alternative x induisant la valeur MmMR courante (c'est-à-dire vérifiant $\text{mMR}(x, \Omega_{\mathcal{P}}) = \text{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}})$), en produisant une question qui implique un de ses ensembles de niveau. Nous proposons alors une stratégie de génération de questions alternative, nommée S1, demandant au décideur de spécifier directement la catégorie qui est la mieux adaptée à l'alternative x selon lui. Pour cette stratégie, les valeurs $\text{mMR}(x, \Omega_{\mathcal{P}})$ sont calculées en optimisant les programmes linéaires non simplifiés (i.e. $\text{LP}_x^{\top}$ et $\text{LP}_x^{\perp}$) en utilisant le solveur Gurobi, puisque les formulations compactes ne sont pas valides avec le type de questions considéré. La figure 2.7 compare

les stratégies S0 et S1 en donnant les moyennes obtenus sur 100 tests.

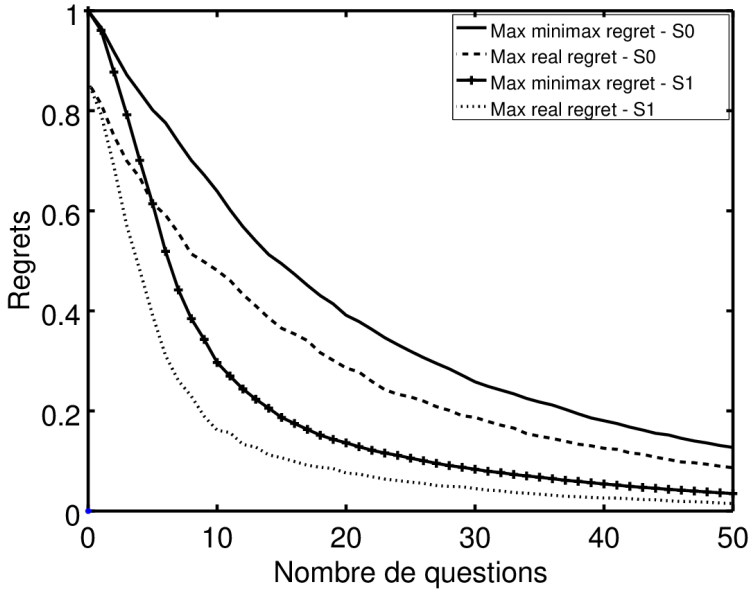


FIGURE 2.7 – Comparaison avec la stratégie S1 demandant de classer l'alternative de plus grand mMR (5 critères, 150 alternatives, 10 catégories).

Sur la figure 2.7, nous pouvons voir que la stratégie S1 est plus informative que la stratégie S0 puisque la valeur MmMR ainsi que le *maximum real regret* diminuent moins rapidement avec cette dernière. Soulignons cependant que la stratégie S1 est praticable uniquement sur des petits problèmes de tri, définis avec suffisamment peu de critères (pour que le nombre de variables et contraintes de monotonie ne soit pas trop grand) et impliquant relativement peu d'alternatives (pour ne pas avoir trop de programmes linéaires à optimiser). Pour les problèmes de tri plus grands, la stratégie S0 est incontestablement la plus appropriée car celle-ci est beaucoup plus rapide que S1 (grâce aux formulations compactes) et réduit la valeur MmMR de manière assez significative.

2.3.3 Méthodes pour le tri avec profils de référence

Dans la sous-section précédente, nous avons étudié l'approche quantitative *agréger puis comparer* (e.g., [Perny, 2000, Grabisch and Perny, 2003]) dans le cadre des problèmes de tri multicritère. Dans la littérature, une autre approche occupe une place prépondérante : le tri avec profils de référence, proposé par Roy dans la méthode Electre TRI (e.g., [Roy, 1981]) et utilisé dans de nombreuses variantes (e.g., [Mousseau and Slowinski, 1998, Perny, 1998, Zopounidis and Doumpos, 2002, Bouyssou and Marchant, 2007a,b]). Cette approche qualitative consiste essentiellement à permuter les étapes d'agrégation et de comparaison, ce qui lui a valu l'appellation *comparer puis agréger* (e.g., [Perny, 2000, Grabisch and Perny, 2003]). Dans cette section, nous nous intéressons plus particulièrement à la méthode de filtrage fondée sur

les préférences introduite dans [Perny, 1998]. Cette méthode utilise un ensemble de vecteurs $\{r^0, r^1, \ldots, r^k\}$ pour délimiter les différentes catégories possibles. Plus précisément, chaque vecteur $r^\ell \in \mathbb{R}^q$ joue le rôle de borne inférieure multicritère de la catégorie K_ℓ, $\ell \in \{1, \ldots, k\}$; le vecteur $r^0 \in \mathbb{R}^q$ est quant à lui choisi pour borner supérieurement tous les critères. Ces vecteurs, appelés profils de référence, représentent le "desiderata" de chaque catégorie en terme de performance sur les critères. Ils sont définis a priori de sorte que, pour tout $\ell \in \{0, \ldots, k-1\}$, nous ayons $r^\ell \gg_P r^{\ell+1}$, i.e. $r_j^\ell > r_j^{\ell+1}$ pour tout $j \in \mathcal{Q} = \{1, \ldots, q\}$ (cf. définition 3). À partir de ces profils, on définit la règle d'affectation suivante : une alternative $x \in \mathcal{X}$ doit être mise dans la catégorie K_ℓ si le décideur préfère x à r^ℓ mais que celui-ci ne préfère pas x à $r^{\ell-1}$. Pour comparer une alternative x et un profil de référence r^ℓ, la méthode de filtrage utilise un index de préférence défini comme suit :

$$P(x, r^\ell, \omega) = f_\omega(P_1(x, r^\ell), \ldots, P_q(x, r^\ell)) \tag{2.46}$$

où f_ω est une fonction d'agrégation paramétrée par ω et $P_j(x, r^\ell)$ est un index de préférence mono-critère défini par :

$$P_j(x, r^\ell) = \begin{cases} 1 & \text{si} \quad x_j - r_j^\ell > \gamma_j^+ \\ \dfrac{x_j - r_j^\ell - \gamma_j^-}{\gamma_j^+ - \gamma_j^-} & \text{si} \quad \gamma_j^- < x_j - r_j^\ell \leq \gamma_j^+ \\ 0 & \text{si} \quad x_j - r_j^\ell \leq \gamma_j^- \end{cases}$$

L'évolution de l'index $P_j(x, r^\ell)$ en fonction de la différence $x_j - r_j^\ell$ est représentée dans la figure 2.8.

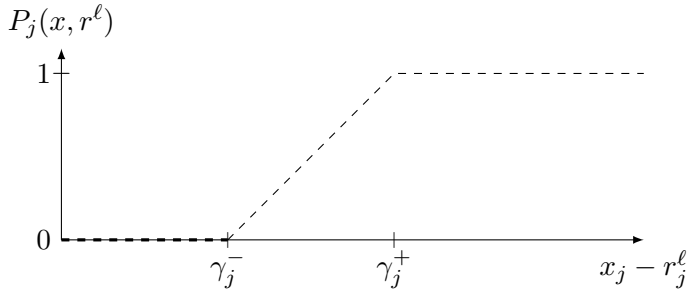


FIGURE 2.8 – $P_j(x, r^\ell)$ en fonction de $x_j - r_j^\ell$.

L'index mono-critère $P_j(x, r^\ell)$ représente la crédibilité de l'information "x est meilleure que r^ℓ sur le critère j". Cet index atteint sa valeur maximale (égale à 1) lorsque la différence de performance $x_j - r_j^\ell$ excède le seuil de préférence γ_j^+. Le seuil de préférence γ_j^+ définit la plus petite valeur $x_j - r_j^\ell$ compatible avec la préférence "x est strictement meilleure que r^ℓ sur le critère j". L'index mono-critère $P_j(x, r^\ell)$ atteint par ailleurs sa valeur minimale (égale à 0) lorsque la différence de performance $x_j - r_j^\ell$ est inférieure au seuil d'indifférence γ_j^-. Le seuil d'indifférence γ_j^- représente la plus grande valeur $x_j - r_j^\ell$ compatible avec une indifférence (ou absence de préférence) entre x et r^ℓ sur le critère j. Entre les deux seuils γ_j^+ et γ_j^-, il existe une zone où le décideur hésite entre la préférence stricte et l'indifférence ; dans cette zone,

l'index mono-critère $P_j(x, r^\ell)$ croît linéairement avec la quantité $x_j - r_j^\ell$ (cf. figure 2.8). Ces seuils font partie intégrante de la définition des échelles d'évaluation des critères et sont préalablement déterminés avec le décideur de sorte que l'inégalité $\gamma_j^+ \geq \gamma_j^-$ soit vraie.

Lorsque f_ω est un opérateur de compromis (i.e. $\min_{j\in\mathcal{Q}}\{x_j\} \leq f_\omega(x) \leq \max_{j\in\mathcal{Q}}\{x_j\}$ pour tout x), l'index $P(x, r^\ell, \omega)$ défini par l'équation (2.46) appartient nécessairement à l'intervalle $[0, 1]$. La valeur 1 est atteinte lorsque les critères sont en faveur de la préférence stricte à l'unanimité, tandis que la valeur 0 est obtenue quand ces derniers sont tous contre la préférence (stricte ou non). Entre ces deux extrêmes, l'index global $P(x, r^\ell, \omega)$ mesure la force des arguments en faveur de la préférence "x est meilleure que r^ℓ". À partir de cette valeur, le degré d'appartenance $d_\ell(x, \omega)$ d'une alternative x à une catégorie K_ℓ, $\ell \in \{1, \ldots, k\}$, est défini de la manière suivante :

$$d_\ell(x, \omega) = \min\Big\{P(x, r^\ell, \omega),\ 1 - P(x, r^{\ell-1}, \omega)\Big\} \tag{2.47}$$

Finalement, x est affectée à la catégorie K_ℓ telle que :

$$d_\ell(x, \omega) = \max_{\ell'\in\{1,\ldots,k\}} d_{\ell'}(x, \omega) \tag{2.48}$$

Si plusieurs catégories K_ℓ vérifient cette propriété, alors le décideur est indifférent entre les différentes affectations correspondantes. L'équation (2.47) garantit par ailleurs que, pour tout alternative x fixée, $d_\ell(x, \omega)$ est une fonction unimodale en ℓ avec une valeur maximale au moins égale à 0.5 (pour plus de détails, voir [Perny, 1998]).

L'usage de profils multicritères r^ℓ au lieu de seuils réels α_ℓ offre la possibilité de raffiner les bornes des catégories pour mieux contrôler le profil des alternatives entrant dans les différentes catégories ; en effet, la catégorie d'une alternative est alors déterminée par son vecteur de performances et non pas par la valeur agrégée de ce dernier. En particulier, deux alternatives ayant la même moyenne mais des vecteurs de performances différents peuvent être affectées à des catégories distinctes, comme illustré dans l'exemple suivant :

Exemple 20. *Considérons un problème de tri avec deux alternatives x, y et deux catégories K_1 et K_2 caractérisées par les profils de référence r^0, r^1 et r^2. Les évaluations multicritères sont les suivantes :*

	1	2	3
x	6	15	15
y	30	3	3
r^0	50	50	50
r^1	12	12	12
r^2	0	0	0

Nous supposons ici que les trois critères considérés sont à maximiser et nous utilisons une échelle d'évaluation commune avec un seuil de préférence $\gamma_j^+ = 1$ et un seuil d'indifférence $\gamma_j^- = 0$, pour tout

critère $j \in \mathcal{Q} = \{1, 2, 3\}$. Par ailleurs, nous supposons que la fonction f_ω utilisée par la méthode de tri est une somme pondérée (cf. définition 11) de poids $\omega = (1/3, 1/3, 1/3)$; autrement dit, $f_\omega(z_{1,2}, z_3) = (z_1 + z_2 + z_3)/3$ pour tout vecteur $z \in \mathbb{R}^3$. Dans ce cas, nous avons $P(x, r^0, \omega) = P(y, r^0, \omega) = 0$ et $P(x, r^2, \omega) = P(y, r^2, \omega) = 1$. La seule différence entre les alternatives x et y provient du profil de référence r^1 : $P(x, r^1, \omega) = 2/3$ et $P(y, r^1, \omega) = 1/3$. Ainsi, en utilisant l'équation (2.47), nous obtenons les degrés d'appartenance suivants :

	d_1	d_2
x	2/3	1/3
y	1/3	2/3

Par conséquent, x est mieux classée que y en utilisant cette méthode de tri puisque x est affectée à la catégorie K_1 alors que y est placée dans la catégorie K_2. Remarquons que ces deux alternatives seraient pourtant indiscernables avec l'approche agréger puis comparer *puisque celles-ci ont exactement la même moyenne : $f_\omega(x) = f_\omega(y)$. Par ailleurs, même si la performance de y passait de 30 à 50 sur le premier critère, cette alternative resterait dans la catégorie K_2 ; l'alternative y aurait alors une meilleure moyenne que x, mais serait moins bien classée que x. Ceci s'explique par le fait suivant : si l'écart de performance entre une alternative z et un profil r^ℓ est supérieur au seuil de préférence sur un critère j, alors toute augmentation de la performance z_j ne peut améliorer l'index de préférence global $P(z, r^\ell, \omega)$; autrement dit, avec cette méthode de tri, des performances mauvaises sur certains critères ne sont pas toujours compensables par de meilleures performances réalisées sur d'autres critères. En ce qui concerne l'alternative y de notre exemple, ses faiblesses sur les critères 2 et 3 ne peuvent pas être compensées par une amélioration de sa performance sur le critère 1.*

Cet exemple permet d'illustrer la nature non-compensatoire de cette méthode de tri, où la différence de score ne joue plus au-delà d'une certaine valeur (contrairement aux méthodes de tri fondées sur les seuils de préférence).

Pour pouvoir mettre en œuvre cette méthode, il convient non seulement de déterminer les profils de référence qui ont un intérêt pour le décideur, mais aussi d'identifier l'instance des paramètres ω permettant de rendre compte de ses préférences. Pour la problématique du tri avec profils de référence, différentes méthodes d'élicitation ont été proposées dans la littérature (e.g., [Mousseau and Slowinski, 1998, Mousseau et al., 2000, Sobrie et al., 2013, 2017b]) et des travaux se sont en particulier intéressés à l'élicitation d'une intégrale de Choquet dans ce cadre (e.g., [Sobrie et al., 2015]). Cependant, à notre connaissance, la problématique du tri n'a pas encore été attaquée sous l'angle de l'élicitation incrémentale fondée sur le concept de regret. Ceci constitue l'objet de cette section.

Dans cette section, nous commençons par proposer une adaptation de la notion de regret pour les problèmes de tri avec profils de référence. Puis, nous montrerons comment les problèmes d'optimisation sous-jacents peuvent être résolus par programmation linéaire lorsque la fonction d'agrégation f_ω est linéaire en ses paramètres ω. Enfin, nous terminons notre étude avec le cas des intégrales de Choquet en réalisant des expériences pour valider notre approche incrémentale.

i) Notions de regret pour le tri avec profils de référence

Pour une instance donnée de ω, il s'agit d'affecter chaque alternative $x \in \mathcal{X}$ à la catégorie K_ℓ vérifiant $d_\ell(x,\omega) \geq d_{\ell'}(x,\omega)$ pour tout $k \in \{1,\ldots,k\}$ (cf. équation (2.48)). De ce fait, il semble naturel de définir le regret $\mathrm{R}(x, K_\ell, K_{\ell'}, \omega)$ associé à l'affectation de x dans la catégorie K_ℓ au lieu de la catégorie $K_{\ell'}$ de la façon suivante :

$$\mathrm{R}(x, K_\ell, K_{\ell'}, \omega) = d_{\ell'}(x,\omega) - d_\ell(x,\omega).$$

Nous supposons ici que l'instance de ω permettant de modéliser au mieux les préférences du décideur est connue de manière imprécise. Notons $\mathcal{P}$ l'ensemble de toutes les données que le système a pu collecter sur les préférences du décideur. Notons ensuite $\Omega_\mathcal{P}$ l'ensemble de toutes les instances de ω compatibles avec les informations contenues dans l'ensemble $\mathcal{P}$. Dans ce contexte, on peut s'intéresser à l'affectation des alternatives permettant de minimiser le regret du décideur dans le pire scénario possible. Pour ce faire, nous introduisons ci-après différentes notions de regret, adaptées au tri avec profils de référence.

Définition 42 (Pairwise Max Regret (PMR))**.** *Le PMR associé à l'affectation de l'alternative $x \in \mathcal{X}$ dans la catégorie K_ℓ au lieu de la catégorie $K_{\ell'}$ est :*

$$\begin{aligned}\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_\mathcal{P}) &= \max_{\omega \in \Omega_\mathcal{P}} \mathrm{R}(x, K_\ell, K_{\ell'}, \omega) \\ &= \max_{\omega \in \Omega_\mathcal{P}} \Big\{ d_{\ell'}(x,\omega) - d_\ell(x,\omega) \Big\}\end{aligned}$$

Par définition, la valeur $\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_\mathcal{P})$ est égale au plus grand écart possible entre les degrés d'appartenance $d_{\ell'}(x,\omega)$ et $d_\ell(x,\omega)$. Cette quantité représente la plus grande erreur que le système puisse commettre en décidant d'affecter x à la catégorie K_ℓ au lieu de la placer dans la catégorie $K_{\ell'}$.

Définition 43 (Max Regret (MR))**.** *The MR associé à l'affectation de x dans la catégorie K_ℓ est :*

$$\mathrm{MR}(x, K_\ell, \Omega_\mathcal{P}) = \max_{\ell' \in \{1,\ldots,k\}} \mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_\mathcal{P})$$

La valeur $\mathrm{MR}(x, K_\ell, \Omega_\mathcal{P})$ représente la plus grande erreur que le système puisse provoquer en affectant l'alternative x à la catégorie K_ℓ (au lieu de la catégorie que le décideur estime convenir le mieux). De ce fait, dans l'optique de minimiser le regret dans le pire scénario possible, il convient d'affecter chaque alternative $x \in \mathcal{X}$ à la catégorie K_ℓ qui minimise cette dernière quantité. Le regret associé à cette décision est défini formellement ci-dessous :

Définition 44 (Minimax Regret (mMR))**.** *Le mMR d'une alternative $x \in \mathcal{X}$ est :*

$$\mathrm{mMR}(x, \Omega_\mathcal{P}) = \min_{\ell \in \{1,\ldots,k\}} \mathrm{MR}(x, K_\ell, \Omega_\mathcal{P})$$

Pour estimer la qualité globale du tri résultant de ces différentes affectations, nous proposons d'agréger les $|\mathcal{X}|$ valeurs $\mathrm{dmMR}(x, \Omega_\mathcal{P})$ comme dans les problèmes de tri avec seuils (cf. Section 2.3.2). Plus précisément, nous allons utiliser le critère suivant :

Définition 45 (Maximum Minimax Regret (MmMR))**.**

$$\mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}}) = \max_{x \in \mathcal{X}} \mathrm{mMR}(x, \Omega_{\mathcal{P}})$$

Lorsque $\mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}}) = 0$, le tri obtenu en minimisant les regrets reflète exactement les exigences du décideur. Dans le cas contraire, si la valeur MmMR est trop élevée, il convient de mettre en œuvre une procédure d'élicitation incrémentale, réduisant progressivement cette valeur, jusqu'à atteindre un niveau de risque que le décideur juge acceptable (quantifié par un seuil de tolérance $\delta \geq 0$). En effet, cette approche est envisageable car nous avons $\Omega_{\mathcal{P}'} \subseteq \Omega_{\mathcal{P}}$ pour tout $\mathcal{P}' \supseteq \mathcal{P}$, ce qui implique :

$$\begin{aligned}
&\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}'}) \leq \mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}}),\ \forall x \in \mathcal{X}, \forall \ell, \ell' \in \{1, \ldots, k\} \\
&\mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}'}) \leq \mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}),\ \forall x \in \mathcal{X}, \forall \ell \in \{1, \ldots, k\} \\
&\mathrm{mMR}(x, \Omega_{\mathcal{P}'}) \leq \mathrm{mMR}(x, \Omega_{\mathcal{P}}),\ \forall x \in \mathcal{X} \\
&\mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}'}) \leq \mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}})
\end{aligned}$$

De ces inégalités, nous déduisons que la valeur $\mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}})$ ne peut augmenter après observation de nouvelles données de préférence ; dans nos tests numériques, nous verrons que cette quantité diminue strictement en pratique lorsque les questions posées au décideur sont choisies consciencieusement.

Comme pour la problématique du choix, il est tout à fait envisageable de demander au décideur de comparer des alternatives. Néanmoins, contrairement aux problèmes de choix, la garantie de performance $\mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}})$ ne suggère pas directement une paire d'alternatives constituant une question de comparaison pertinente ; nous rencontrons en fait le même problème que dans la problématique de tri avec seuils de préférence (cf. Section 2.3.2). À la place, nous allons demander au décideur de choisir, parmi un ensemble de catégories, celle qui lui semble être la plus adéquate pour une alternative donnée. Pour ce type de questions, nous noterons $(x, K_\ell \succsim K_{\ell'})$ la donnée "K_ℓ est meilleure que $K_{\ell'}$ pour x". Afin de choisir la prochaine question à poser, nous pouvons théoriquement utiliser le critère de sélection WMmMR (cf. définition 41). Cependant, l'usage de ce critère semble difficile en pratique car le nombre de questions possibles augmente ici avec le nombre d'alternatives et de catégories du problème. Dans le même esprit que la stratégie CSS, conçue pour les problèmes de choix (cf. Section 1.4.2), nous proposons de nous concentrer sur l'alternative x maximisant la valeur mMR. En effet, en faisant ce choix, nous augmentons nos chances de réduire la valeur MmMR courante puisque $\mathrm{MmMR}(\mathcal{X}, \Omega_{\mathcal{P}}) = \mathrm{mMR}(x, \Omega_{\mathcal{P}})$. Nous envisageons alors les deux stratégies suivantes :

- Demander au décideur de placer directement l'alternative x dans la bonne catégorie.
- Demander au décideur de comparer deux catégories K_ℓ et $K_{\ell'}$ qui vérifient $\mathrm{mMR}(x, \Omega_{\mathcal{P}}) = \mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) = \mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}})$.

Cette dernière approche présente l'avantage de poser des questions plus simples (comparaison de seulement deux catégories au lieu de toutes), tout en se concentrant sur une paire de catégories induisant la valeur MmMR courante. Soulignons que, pour la problématique du tri avec seuils de préférence, l'identi-

fication d'une telle paire est moins directe, puisque la notion de regret entre deux catégories (PMR) n'a pas été définie dans ce contexte (cf. Section 2.3.2).

ii) Détermination du tri optimal au sens du critère MmMR

Comme pour les problèmes de choix, la détermination des valeurs PMR représente la principale difficulté liée au calcul des valeurs MmMR dans le cadre de la problématique du tri avec profils de référence. Calculer la valeur $\text{PMR}(x, K_\ell, K_{\ell'}, \Omega_\mathcal{Q})$ est ici assez complexe puisqu'il s'agit de maximiser une différence de minimums. Néanmoins, nous allons montrer que cette difficulté peut être contournée en décomposant ce problème d'optimisation en deux sous-problèmes plus simples, pouvant être modélisés par des programmes linéaires en nombres entiers (lorsque f_ω est linéaire en ω). Pour ce faire, nous allons commencer par démontrer que les réponses du décideur (stockées dans $\mathcal{P}$) induisent uniquement des contraintes linéaires sur l'espace des paramètres.

Proposition 14. *Pour toute alternative $x \in \mathcal{X}$ et toutes catégories $K_\ell, K_{\ell'}$, avec $\ell \neq \ell'$:*

$$\Omega_{\mathcal{P}\cup(x,K_\ell \succsim K_{\ell'})} = \begin{cases} \Big\{\omega \in \Omega_\mathcal{P} : 1 - P(x, r^{\ell-1}, \omega) \geq \min\{P(x, r^{\ell'}, \omega), 1 - P(x, r^{\ell'-1}, \omega)\}\Big\} & \textit{si } \ell > \ell' \\ \Big\{\omega \in \Omega_\mathcal{P} : P(x, r^{\ell}, \omega) \geq \min\{P(x, r^{\ell'}, \omega), 1 - P(x, r^{\ell'-1}, \omega)\}\Big\} & \textit{sinon} \end{cases}$$

Démonstration. Si le décideur nous informe que la catégorie K_ℓ est plus adaptée que la catégorie $K_{\ell'}$ pour l'alternative x, nous devons restreindre l'ensemble $\Omega_\mathcal{P}$ à toutes les paramètres ω vérifiant $d_\ell(x, \omega) \geq d_{\ell'}(x, \omega)$, c'est-à-dire :

$$\min\Big\{P(x, r^\ell, \omega),\ 1 - P(x, r^{\ell-1}, \omega)\Big\} \geq \min\Big\{P(x, r^{\ell'}, \omega),\ 1 - P(x, r^{\ell'-1}, \omega)\Big\}$$

Notons que cette restriction revient à imposer les contraintes suivantes :

$$P(x, r^\ell, \omega) \geq \min\Big\{P(x, r^{\ell'}, \omega),\ 1 - P(x, r^{\ell'-1}, \omega)\Big\} \tag{2.49}$$

$$1 - P(x, r^{\ell-1}, \omega) \geq \min\Big\{P(x, r^{\ell'}, \omega),\ 1 - P(x, r^{\ell'-1}, \omega)\Big\} \tag{2.50}$$

Tout d'abord, considérons le cas où l'inégalité $\ell > \ell'$ est vérifiée. Dans ce cas, comme $r^{\ell'} \gg_P r^\ell$ par définition, alors nous avons forcément $P_j(x, r^\ell) \geq P_j(x, r^{\ell'})$ pour tout critère $j \in \mathcal{Q}$, c'est-à-dire $(P_1(x, r^\ell), \ldots, P_q(x, r^\ell)) \succsim_P (P_1(x, r^{\ell'}), \ldots, P_q(x, r^{\ell'}))$. Si par ailleurs f_ω est un opérateur monotone (ce qui est le cas de tous les agrégateurs que nous considérons dans cette thèse), alors nous en déduisons $P(x, r^\ell, \omega) \geq P(x, r^{\ell'}, \omega)$ pour tout $\omega \in \Omega_\mathcal{P}$. Par conséquent, la contrainte (2.49) est satisfaite par tout $\omega \in \Omega_\mathcal{P}$, et donc rien ne sert de l'imposer après insertion de la préférence $(x, K_\ell \succsim K_{\ell'})$ dans $\mathcal{P}$. De ce fait, en considérant l'équation (2.50), nous avons bien :

$$\Omega_{\mathcal{P}\cup(x,K_\ell \succsim K_{\ell'})} = \Big\{\omega \in \Omega_\mathcal{P} : 1 - P(x, r^{\ell-1}, \omega) \geq \min\{P(x, r^{\ell'}, \omega),\ 1 - P(x, r^{\ell'-1}, \omega)\}\Big\}$$

À présent, considérons le cas où c'est l'inégalité $\ell < \ell'$ qui est vraie. Dans ce cas, nous avons $P_j(x, r^{\ell-1}) \leq P_j(x, r^{\ell'-1})$ pour tout $j \in \mathcal{Q}$ car $r^{\ell-1} \gg_P r^{\ell'-1}$ par définition. Par conséquent, si f_ω est monotone, alors nous avons $P(x, r^{\ell-1}, \omega) \leq P(x, r^{\ell'-1}, \omega)$ pour tout $\omega \in \Omega_\mathcal{P}$, ce qui implique $1 - P(x, r^{\ell-1}, \omega) \geq 1 - P(x, r^{\ell'-1}, \omega)$ pour tout $\omega \in \Omega_\mathcal{P}$. Ceci montre que la contrainte (2.50) est satisfaite pour tout $\omega \in \Omega_\mathcal{P}$, et donc nous avons dans ce cas :

$$\Omega_{\mathcal{P} \cup (x, K_\ell \succsim K_{\ell'})} = \Big\{ \omega \in \Omega_\mathcal{P} : P(x, r^\ell, \omega) \geq \min\{P(x, r^{\ell'}, \omega),\ 1 - P(x, r^{\ell'-1}, \omega)\} \Big\} \qquad \square$$

Cette proposition montre que, si le décideur nous transmet une donnée de type $(x, K_\ell \succsim K_{\ell'})$, alors il suffit d'imposer une seule contrainte sur l'espace des paramètres :

$$\begin{cases} 1 - P(x, r^{\ell-1}, \omega) \geq \min \Big\{ P(x, r^{\ell'}, \omega),\ 1 - P(x, r^{\ell'-1}, \omega) \Big\} & \text{si } \ell > \ell' \\ P(x, r^\ell, \omega) \geq \min \Big\{ P(x, r^k, \omega),\ 1 - P(x, r^{k-1}, \omega) \Big\} & \text{sinon} \end{cases}$$

Bien que ces inégalités ne soient pas linéaires en ω à cause de l'opérateur minimum, nous pouvons les rendre linéaires en appliquant des techniques standards de linéarisation de cet opérateur. Par exemple, dans le cas où $\ell > \ell'$, il suffit d'imposer les deux contraintes suivantes :

$$\begin{cases} Mb + 1 - P(x, r^{\ell-1}, \omega) \geq P(x, r^{\ell'}, \omega) \\ M(1-b) + 1 - P(x, r^{\ell-1}, \omega) \geq 1 - P(x, r^{\ell'-1}, \omega) \end{cases}$$

où b est une variable booléenne et M est une constante plus grande que 1. Pour nous rendre compte que l'introduction de cette variable booléenne permet de contraindre de manière équivalente l'espace des paramètres, nous pouvons étudier ce qu'il se passe dans les deux scénarios :

- si $b = 0$, alors la seconde contrainte n'a plus aucun effet puisque $M \geq 1$ et $0 \leq P(x, r^{\ell''}, \omega) \leq 1$ pour tout $\ell'' \in \{1, \ldots, k\}$ et tout $\omega \in \Omega_\mathcal{P}$. Autrement dit, dans ce scénario, seule la première contrainte est imposée : $1 - P(x, r^{\ell-1}, \omega) \geq P(x, r^{\ell'}, \omega)$.
- si $b = 1$, alors c'est la première contrainte qui n'a plus aucun effet (pour les mêmes raisons que le cas précédent), et donc seule la seconde contrainte est imposée : $1 - P(x, r^{\ell-1}, \omega) \geq 1 - P(x, r^{\ell'-1}, \omega)$.

Ainsi, la valeur $1 - P(x, r^{\ell-1}, \omega)$ doit être supérieure à $P(x, r^{\ell'}, \omega)$ ou $1 - P(x, r^{\ell'-1}, \omega)$, en fonction de la valeur de b. Par conséquent, si la variable b n'apparaît dans aucune autre contrainte, alors tout algorithme d'optimisation préférera choisir, pour tout $\omega \in \Omega_\mathcal{P}$, la valeur de b telle que la quantité $1 - P(x, r^{\ell'-1}, \omega)$ est la moins contrainte possible. Par conséquent, pour tout $\omega \in \Omega_\mathcal{P}$, la valeur de b sera choisie pour que la quantité $1 - P(x, r^{\ell'-1}, \omega)$ doivent être plus grande que le minimum entre $P(x, r^{\ell'}, \omega)$ et $1 - P(x, r^{\ell'-1}, \omega)$; ceci modélise exactement la contrainte que nous souhaitons imposer quand $\ell > \ell'$. Similairement, dans le cas où $\ell < \ell'$, il nous suffit d'imposer :

$$\begin{cases} Mb + P(x, r^\ell, \omega) \geq P(x, r^{\ell'}, \omega) \\ M(1-b) + P(x, r^\ell, \omega) \geq 1 - P(x, r^{\ell'-1}, \omega) \end{cases}$$

où b est une variable booléenne et M est une constante plus grande que 1. Ainsi, nous venons de prouver que l'ensemble $\Omega_{\mathcal{P}}$ peut être décrit avec uniquement des contraintes linéaires. Ce résultat est utilisé dans la proposition suivante :

Proposition 15. *Pour toute alternative $x \in \mathcal{X}$ et toutes catégories $K_\ell, K_{\ell'}$, avec $\ell \neq \ell'$, nous avons :*

$$\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}}) = \max\{\beta_1, \beta_2\}$$

où β_1 et β_2 sont respectivement les valeurs optimales des programmes linéaires en nombres entiers :

$$\begin{array}{ll} \max\limits_{\omega \in \Omega_{\mathcal{P}}} & \Big\{t - P(x, r^{\ell}, \omega)\Big\} \\ s.c. & t \leq P(x, r^{\ell'}, \omega) \\ & t \leq 1 - P(x, r^{\ell'-1}, \omega) \\ & t \geq 0 \end{array} \qquad \begin{array}{ll} \max\limits_{\omega \in \Omega_{\mathcal{P}}} & \Big\{t + P(x, r^{\ell-1}, \omega) - 1\Big\} \\ s.c. & t \leq P(x, r^{\ell'}, \omega) \\ & t \leq 1 - P(x, r^{\ell'-1}, \omega) \\ & t \geq 0 \end{array}$$

Démonstration. Par définition, pour tout $x \in \mathcal{X}$ et toutes catégories $K_\ell, K_{\ell'}$, $\ell \neq \ell'$, nous avons :

$$\begin{aligned} \mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}}) &= \max_{\omega \in \Omega_{\mathcal{P}}} \mathrm{R}(x, K_\ell, K_{\ell'}, \omega) \\ &= \max_{\omega \in \Omega_{\mathcal{P}}} \Big\{d_{\ell'}(x, \omega) - d_\ell(x, \omega)\Big\} \\ &= \max_{\omega \in \Omega_{\mathcal{P}}} \Big\{ \min\{P(x, r^{\ell'}, \omega),\, 1 - P(x, r^{\ell'-1}, \omega)\} - d_\ell(x, \omega)\Big\} \end{aligned}$$

Pour linéariser la fonction à maximiser, nous remplaçons la quantité $\min\{P(x, r^{\ell'}, \omega), 1 - P(x, r^{\ell'-1}, \omega)\}$ par une variable réelle t positive contrainte de vérifier :

$$\begin{aligned} & t \leq P(x, r^{\ell'}, \omega) \\ & t \leq 1 - P(x, r^{\ell'-1}, \omega) \end{aligned}$$

ce qui constitue une application directe des techniques classiques de linéarisation de l'opérateur minimum (lorsque que celui-ci est présent dans la fonction objectif). En résumé, nous pouvons calculer $\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}})$ en optimisant le programme suivant :

$$\begin{array}{ll} \max\limits_{\omega \in \Omega_{\mathcal{P}}} & \Big\{t - d_\ell(x, \omega)\Big\} \\ \text{s.c.} & t \leq P(x, r^{\ell'}, \omega) \\ & t \leq 1 - P(x, r^{\ell'-1}, \omega) \\ & t \geq 0 \end{array}$$

Puis, comme nous avons :

$$\begin{aligned} t - d_\ell(x,\omega) &= t - \min\{P(x,r^\ell,\omega),\ 1 - P(x,r^{\ell-1},\omega)\} \\ &= t + \max\{-P(x,r^\ell,\omega),\ P(x,r^{\ell-1},\omega) - 1\} \\ &= \max\{t - P(x,r^\ell,\omega),\ t + P(x,r^{\ell-1},\omega) - 1\} \end{aligned}$$

alors nous pouvons réécrire le programme précédent de la manière suivante :

$$\begin{aligned} \max_{\omega\in\Omega_\mathcal{P}} \quad & \Big\{ \max\{t - P(x,r^\ell,\omega),\ t + P(x,r^{\ell-1},\omega) - 1\} \Big\} \\ \text{s.c.} \quad & t \leq P(x,r^{\ell'},\omega) \\ & t \leq 1 - P(x,r^{\ell'-1},\omega) \\ & t \geq 0 \end{aligned}$$

Le résultat désiré peut finalement être obtenu en permutant les deux opérateurs maximum. □

Cette proposition nous offre la possibilité de calculer $\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_\mathcal{P})$ en optimisant deux programmes linéaires en nombres entiers, la quantité recherchée correspondant alors à la plus grande des deux valeurs optimales. Dans la sous-section suivante, nous allons voir que ce calcul peut être davantage simplifié lorsque f_ω est une intégrale de Choquet définie à partir d'une capacité strictement monotone.

iii) Application aux intégrales de Choquet

Dans cette sous-section, nous considérons une instance particulière, obtenue en interprétant f_ω comme une intégrale de Choquet (cf. définition 16). Cet agrégateur permet ici de modéliser des synergies entre les différents critères lors de l'agrégation des index $P_j(x, r^\ell)$ en un index global $P(x, r^\ell, \omega)$. Dans ce contexte, la capacité v joue le rôle des paramètres ω, l'ensemble des paramètres réalisables $\Omega_\mathcal{P}$ étant alors composé de toutes les capacités normalisées (cf. définition 13) compatibles avec les données de préférence dans $\mathcal{P}$. Comparer une alternative $x \in \mathcal{X}$ à un profil de référence r^ℓ se fait donc à travers l'index de préférence :

$$P(x, r^\ell, v) = \mathrm{Ch}((P_1(x, r^\ell), \ldots, P_q(x, r^\ell)), v)$$

Le degré d'appartenance de x à la catégorie K_ℓ est quant à lui défini par :

$$d_\ell(x, v) = \min\Big\{P(x, r^\ell, v),\ 1 - P(x, r^{\ell-1}, v)\Big\}$$

Comme l'intégrale de Choquet est une fonction linéaire en v à x fixé, alors nous pouvons calculer $\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_\mathcal{P})$ par programmation linéaire en nombres entiers en utilisant la proposition 15. Néanmoins, dans le cas particulier où les capacités possibles sont strictement monotones (c'est-à-dire $v(A) < v(B)$ pour tout $A \subset B \subseteq \mathcal{Q}$), nous allons montrer qu'il est possible d'obtenir une linéarisation des contraintes de compatibilité avec $\mathcal{P}$ ne faisant intervenir aucune variable booléenne.

Proposition 16. *Pour toute alternative $x \in \mathcal{X}$ et toutes catégories $K_\ell, K_{\ell'}$, avec $\ell \neq \ell'$:*

$$\Omega_{\mathcal{P}\cup(x,K_\ell \succsim K_{\ell'})} = \begin{cases} \Big\{v \in \Omega_{\mathcal{P}} : 1 - P(x, r^{\ell-1}, v) \geq P(x, r^{\ell'}, v)\Big\} & \text{si } \ell > \ell' \text{ et } \exists j \in \mathcal{Q}, P_j(x, r^{\ell-1}) \neq P_j(x, r^{\ell'-1}) \\ \Big\{v \in \Omega_{\mathcal{P}} : P(x, r^{\ell}, v) \geq 1 - P(x, r^{\ell'-1}, v)\Big\} & \text{si } \ell < \ell' \text{ et } \exists j \in \mathcal{Q},\ P_j(x, r^{\ell}) \neq P_j(x, r^{\ell'}) \\ \Omega_{\mathcal{P}} & \text{sinon} \end{cases}$$

Démonstration. Considérons tout d'abord le cas où $\ell > \ell'$. D'après la proposition 14, les capacités $v \in \Omega_{\mathcal{P}}$ satisfaisant la préférence $(x, K_\ell \succsim K_{\ell'})$ sont précisément celles qui vérifient l'inégalité suivante :

$$1 - P(x, r^{\ell-1}, v) \geq \min\{P(x, r^{\ell'}, v), 1 - P(x, r^{\ell'-1}, v)\} \tag{2.51}$$

Comme $\ell > \ell'$ par hypothèse, alors nous avons $r^{\ell'} \gg_P r^{\ell}$ par définition des profils de référence. Ce dernier fait implique que nous avons $P_j(x, r^{\ell-1}) \geq P_j(x, r^{\ell'-1})$ pour tout $j \in \mathcal{Q}$. Supposons tout d'abord que $P_j(x, r^{\ell-1}) = P_j(x, r^{\ell'-1})$ pour tout $j \in \mathcal{Q}$. Dans ce cas, pour toute capacité v, nous avons $P(x, r^{\ell-1}, v) = P(x, r^{\ell'-1}, v)$, ce qui implique $1 - P(x, r^{\ell-1}, v) = 1 - P(x, r^{\ell'-1}, v)$. Ainsi, toutes les capacités $v \in \Omega_{\mathcal{P}}$ vérifient l'équation (2.51), ce qui nous permet de déduire $\Omega_{\mathcal{P}\cup(x,K_\ell \succsim K_{\ell'})} = \Omega_{\mathcal{P}}$. Supposons maintenant que nous avons $P_j(x, r^{\ell-1}) \neq P_j(x, r^{\ell'-1})$ pour au moins un critère $j \in \mathcal{Q}$. Dans ce cas, nous avons en fait $(P_1(x, r^{\ell-1}), \ldots, P_q(x, r^{\ell-1})) \succ_P (P_1(x, r^{\ell'-1}), \ldots, P_q(x, r^{\ell'-1}))$. Comme par ailleurs l'intégrale de Choquet est un agrégateur strictement monotone quand v est strictement monotone, alors nous en déduisons $P(x, r^{\ell-1}, v) > P(x, r^{\ell'-1}, v)$ pour tout $v \in \Omega_{\mathcal{P}}$, ce qui implique $1 - P(x, r^{\ell-1}, v) < 1 - P(x, r^{\ell'-1}, v)$ pour tout $v \in \Omega_{\mathcal{P}}$. Ainsi, l'inégalité (2.51) est vraie pour une capacité $v \in \Omega_{\mathcal{P}}$ donnée si et seulement si l'inégalité $1 - P(x, r^{\ell-1}, v) \geq P(x, r^{\ell'}, v)$ est vraie. De ce fait, nous avons dans ce cas $\Omega_{\mathcal{P}\cup(x,K_\ell \succsim K_{\ell'})} = \{v \in \Omega_{\mathcal{P}} : 1 - P(x, r^{\ell-1}, v) \geq P(x, r^{\ell'}, v)\}$.

Considérons maintenant le cas où $\ell < \ell'$. Dans ce cas, d'après la proposition 14, les capacités $v \in \Omega_{\mathcal{P}}$ satisfaisant la préférence $(x, K_\ell \succsim K_{\ell'})$ sont exactement celles qui vérifient :

$$P(x, r^{\ell}, v) \geq \min\{P(x, r^{\ell'}, v),\ 1 - P(x, r^{\ell'-1}, v)\} \tag{2.52}$$

Puisque l'inégalité $\ell < \ell'$ est vraie par hypothèse, alors nous avons $P_j(x, r^{\ell}) \leq P_j(x, r^{\ell'})$ pour tout critère $j \in \mathcal{Q}$. Supposons tout d'abord que l'égalité $P_j(x, r^{\ell}) = P_j(x, r^{\ell'})$ est vraie pour tout $j \in \mathcal{Q}$. Dans ce cas, nous avons $P(x, r^{\ell}, v) = P(x, r^{\ell'}, v)$ pour toute capacité v, et donc l'équation (2.52) est forcément vérifiée par toutes les capacités $v \in \Omega_{\mathcal{P}}$. Par conséquent, nous avons $\Omega_{\mathcal{P}\cup(x,K_\ell \succsim K_{\ell'})} = \Omega_{\mathcal{P}}$ dans ce cas. À présent, supposons qu'il existe un critère $j \in \mathcal{Q}$ tel que $P_j(x, r^{\ell}) \neq P_j(x, r^{\ell'})$. Dans ce cas, nous avons en réalité $(P_1(x, r^{\ell'}), \ldots, P_q(x, r^{\ell'})) \succ_P P_1(x, r^{\ell}), \ldots, P_q(x, r^{\ell}))$. Puisque l'intégrale de Choquet est un opérateur strictement monotone quand la capacité v est strictement monotone, alors nous avons nécessairement $P(x, r^{\ell}, v) < P(x, r^{\ell'}, v)$ pour toute capacité $v \in \Omega_{\mathcal{P}}$. Ce dernier fait nous apprend que l'inégalité (2.52) est vraie pour une capacité $v \in \Omega_{\mathcal{P}}$ donnée si et seulement si nous avons $P(x, r^{\ell}, v) \geq 1 - P(x, r^{\ell'-1}, v)$. Ceci nous permet de dériver l'égalité suivante : $\Omega_{\mathcal{P}\cup(x,K_\ell \succsim K_{\ell'})} = \{v \in \Omega_{\mathcal{P}} : P(x, r^{\ell}, v) \geq 1 - P(x, r^{\ell'-1}, v)\}$. □

D'après la proposition 16, si le décideur nous transmet la préférence $(x, K_\ell, K_{\ell'})$, alors la mise à jour de l'espace des paramètres peut être réalisée en imposant au plus une contrainte linéaire :

- $1 - P(x, r^{\ell-1}, v) \geq P(x, r^{\ell'}, v)$ si $\ell > \ell'$ et $P_j(x, r^{\ell-1}) \neq P_j(x, r^{\ell'-1})$ pour au moins un $j \in \mathcal{Q}$,
- $P(x, r^{\ell}, v) \geq 1 - P(x, r^{\ell'-1}, v)$ si $\ell < \ell'$ et $P_j(x, r^{\ell}) \neq P_j(x, r^{\ell'})$ pour au moins un $j \in \mathcal{Q}$ et
- aucune contrainte sinon.

De ce fait, l'ensemble $\mathcal{P}$ peut être entièrement décrit par des contraintes linéaires sur des variables réelles. Ainsi, lorsque les préférences du décideur sont représentables par une intégrale de Choquet dont la capacité est strictement monotone, nous pouvons calculer la valeur $\mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}})$ par programmation linéaire en utilisant la proposition 15.

Soulignons toutefois que, pour l'intégrale de Choquet, le nombre de variables et de contraintes impliquées dans ce calcul est exponentiel en le nombre de critères. Notons alors que, dans le cas particulier où $\gamma_j^- = \gamma_j^+$ pour tout $j \in \mathcal{Q}$ (c'est-à-dire $P_j(x, r^\ell) = 1$ si $x_j > r_j^\ell$ et $P_j(x, r^\ell) = 0$ sinon), la contrainte induite par une donnée de type $(x, K_\ell \succsim K_{\ell'})$ est soit de la forme $1 - v(A) \geq v(B)$ soit de la forme $v(A) \geq 1 - v(B)$, avec $A, B \subseteq \mathcal{Q}$. En conséquence, dans ce cas particulier, nous pouvons réduire le nombre de variables et de contraintes en utilisant la proposition 4. Plus précisément, toutes les variables qui n'apparaissent ni dans la fonction objectif, ni dans les contraintes de compatibilité avec $\mathcal{P}$, peuvent être retirées des programmes linéaires à optimiser. Ceci nous conduit à des formulations comprenant au plus $2(|\mathcal{P}| + q) - 1$ variables et des contraintes de monotonie entre ces dernières uniquement. Dans le cas général, ces programmes linéaires restent toutefois relativement simples à optimiser avec un solveur actuel pourvu que le nombre de critères est raisonnablement petit.

iv) Résultats expérimentaux

Dans nos expériences, le profil multicritère de chaque alternative est engendré aléatoirement et uniformément dans $[0, 1]^q$. Par ailleurs, les profils de référence que nous considérons sont sous la forme de profils d'utilités constants, divisant l'échelle d'utilité $[0, 1]$ en intervalles de même taille sur chaque critère. Dans cette sous-section, nous commençons par comparer les performances des différentes stratégies de génération de questions suivantes :

- **S0** : cette stratégie d'élicitation consiste à demander au décideur de nous indiquer la catégorie d'une alternative choisie aléatoirement.
- **S1** : cette stratégie se concentre sur une alternative x maximisant la valeur mMR en demandant au décideur si la catégorie K_ℓ est plus adaptée que la catégorie $K_{\ell'}$ pour x, où $K_\ell, K_{\ell'}$ sont choisies telles que $\mathrm{mMR}(x, \Omega_{\mathcal{P}}) = \mathrm{MR}(x, K_\ell, \Omega_{\mathcal{P}}) = \mathrm{PMR}(x, K_\ell, K_{\ell'}, \Omega_{\mathcal{P}})$.
- **S2** : cette stratégie se concentre aussi sur une alternative maximisant la valeur mMR, mais demande au décideur de nous indiquer directement la catégorie de cette alternative.

Les réponses aux questions sont ici simulées en utilisant une intégrale de Choquet dont la capacité a été engendrée aléatoirement. Sur les figures 2.9 et 2.10, nous reportons respectivement la valeur MmMR et le *maximum real regret* (calculé avec l'intégrale de Choquet du décideur) observés à chaque itération de ces procédures de tri interactives ; les résultats présentés correspondent à une moyenne sur 100 tests.

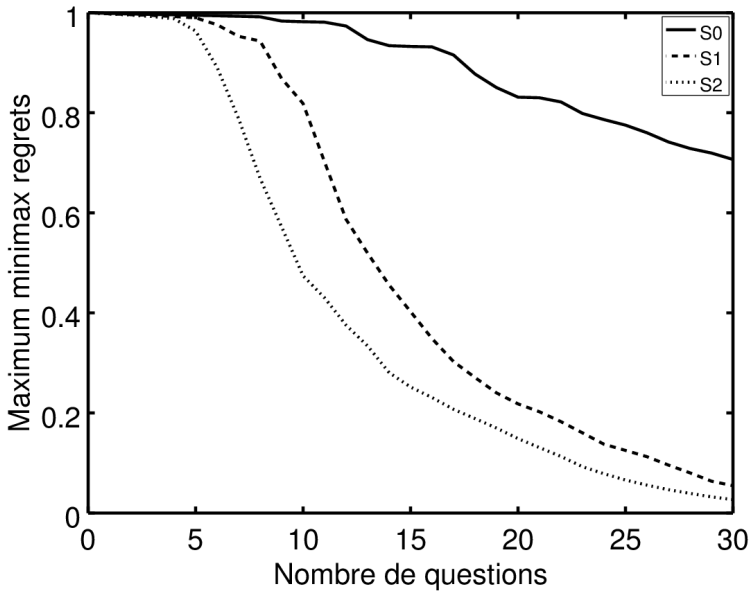


FIGURE 2.9 – Évolution de la valeur MmMR pour les stratégies S0, S1 et S2 (5 critères, 150 alternatives, 5 catégories).

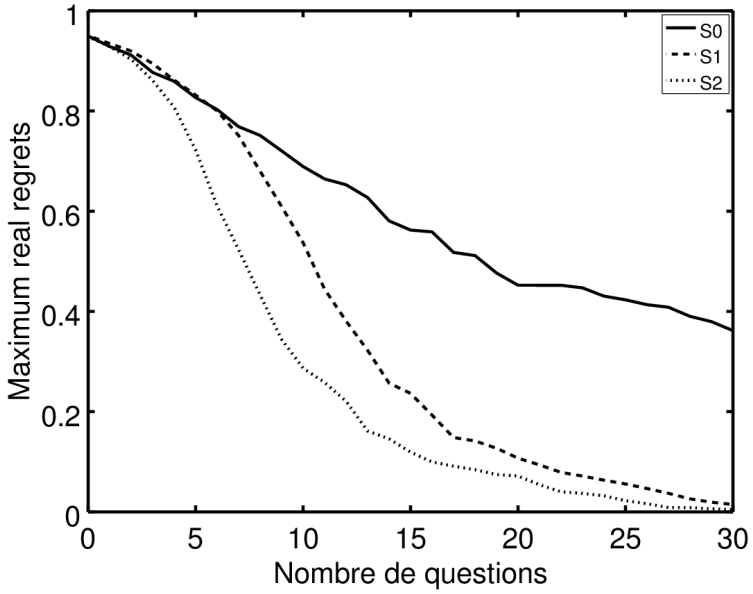


FIGURE 2.10 – Évolution du *maximum real regret* pour les stratégies S0, S1 et S2 (5 critères, 150 alternatives, 5 catégories).

Tout d'abord, nous pouvons voir que la valeur MmMR réduit beaucoup plus rapidement avec les stratégies S1 et S2 qu'avec la stratégie aléatoire S0 ; par exemple, après 20 questions en moyenne, la

valeur MmMR est en dessous de 30% du regret initial avec S1 et S2, alors que cette quantité est toujours au dessus de 80% du regret initial avec la stratégie aléatoire S0. Nous observons la même tendance avec les *maximum real regrets*. Par ailleurs, comme nous pouvions nous y attendre, la stratégie S2 se montre plus informative que la stratégie S1 en pratique. En effet, la valeur MmMR ainsi que le *maximum real regret* réduisent plus rapidement avec cette première que cette dernière. Cependant, nous pouvons aussi constater que S1 parvient à la même valeur MmMR que S2 en posant approximativement trois questions supplémentaires. Cette observation nous suggère d'utiliser S1 plutôt que S2, puisque les questions engendrées par cette dernière sont un peu plus compliquées que celles posées par la première. En effet, il semble plus facile de comparer deux catégories pour une alternative donnée que de déterminer la catégorie adéquate pour cette alternative (car ceci nécessite de comparer implicitement toutes les catégories possibles).

À présent, nous cherchons à évaluer l'impact des paramètres suivants sur les temps de calcul de la valeur MmMR pour la stratégie S1 : k le nombre de catégories, q le nombre de critères, p le nombre de données de préférence disponibles et $|\mathcal{X}|$ le nombre d'alternatives à trier. Pour résoudre les problèmes d'optimisation liés au calcul des PMR (cf. proposition 15), nous faisons appel au solveur de programme linéaire Gurobi depuis un programme écrit en Java. La table 2.3 nous donne les moyennes des temps obtenus sur 100 tests réalisés sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire.

Table 2.3 – Temps de calcul (en secondes) pour déterminer MmMR.

	$k=5$				$k=10$			
	$q=5$		$q=7$		$q=5$		$q=7$	
	$p=0$	$p=5$	$p=0$	$p=5$	$p=0$	$p=5$	$p=0$	$p=5$
$\|\mathcal{X}\|=50$	0.8	1.4	1.8	2.3	3.5	5.6	6.7	8.6
$\|\mathcal{X}\|=100$	1.1	1.8	2.1	3.3	4.6	6.8	9.1	10.8
$\|\mathcal{X}\|=200$	2.2	3.5	6.9	7.1	7.6	10.7	18.2	21.1

Dans cette table, nous voyons en particulier que les temps de résolution sont grandement impactés par la valeur k, correspondant au nombre de catégories. Ceci s'explique par le fait que le nombre de PMR à optimiser est dans le pire cas quadratique en ce dernier paramètre. Par ailleurs, les temps de calcul augmentent significativement avec la valeur q, représentant le nombre de critères du problème. Ceci est essentiellement dû au fait que le nombre de variables et de contraintes de monotonie présentes dans les formulations linéaires permettant de calculer les valeurs PMR est exponentiel en le nombre de critères.

2.4 Conclusion du chapitre

L'élicitation incrémentale fondée sur le critère Minimax Regret (e.g., [Boutilier et al., 2006]) est une approche permettant de formuler des recommandations en présence de préférences imprécisément définies avec garantie de performance par rapport au pire scénario possible. Rappelons que les méthodes

incrémentales arrêtent de poser des questions dès que l'erreur dans le pire scénario ne dépasse pas un certain seuil de tolérance que le décideur juge acceptable. Ceci permet de gagner du temps et de limiter le nombre de questions, en comparaison avec les approches visant à déterminer précisément la fonction d'utilité du décideur. Selon nous, ceci constitue la particularité et la contribution de l'approche incrémentale pour l'aide à la décision.

Dans ce chapitre, nous avons proposé des stratégies efficaces pour l'élicitation incrémentale d'une intégrale de Choquet dans le cadre des problématiques de choix, de rangement et de tri. Le calcul de *minimax regrets* avec une intégrale de Choquet pose un certain nombre de difficultés, principalement liées au nombre de paramètres de ce modèle décisionnel et au nombre de contraintes nécessaires à la caractérisation des paramètres admissibles. Néanmoins, nous sommes parvenus à montrer que les *minimax regrets* peuvent être calculés de manière efficace, à condition de restreindre la forme des questions posées au décideur durant l'élicitation. En particulier, pour la problématique du choix, nous avons tout d'abord prouvé que les *minimax regrets* peuvent être obtenus en temps polynomial par programmation linéaire, puis nous avons proposé un algorithme itératif permettant de calculer ces valeurs encore plus rapidement. Par ailleurs, nous avons présenté des tests numériques permettant de valider nos méthodes expérimentalement en terme de temps de résolution, nombre de questions posées et qualité de la recommandation.

Nos travaux se distinguent de ceux qui existent sur l'intégrale de Choquet en se concentrant sur l'approche incrémentale et en permettant de formuler des recommandations sans avoir à faire des hypothèses restrictives sur l'intégrale de Choquet a priori. En particulier, il a été récemment proposé de choisir l'intégrale de Choquet maximisant la marge des contraintes induites par les données de préférence disponibles (de manière similaire au classificateurs SVM) [Ah-Pine et al., 2013]. Ce genre d'estimation par points ignore la particularité des alternatives du problème, contrairement à l'élicitation fondée sur la notion de regret qui se concentre sur la partie utile de la fonction d'utilité en posant des questions pour discriminer entre les alternatives. Par ailleurs, la méthode proposée ne fournit pas directement une stratégie de génération de questions qui faciliterait son intégration dans une méthode incrémentale.

Les méthodes que nous avons proposées dans ce chapitre ne sont néanmoins pas directement applicables dans des situations décisionnelles où les alternatives sont en très grand nombre (comme en optimisation combinatoire). En effet, dans ces situations, il ne semble pas raisonnable d'énumérer toutes les paires de solutions possibles et de calculer la valeur PMR associée à chacune de ces paires. Pour pouvoir faire face à cette difficulté, nous étudions dans le chapitre suivant le potentiel d'une nouvelle approche, consistant à entremêler élicitation des préférences et résolution du problème de décision. Il s'agit plus précisément de poser des questions au décideur à des moments clés de la résolution, de sorte à limiter à la fois le nombre d'interactions avec celui-ci et les temps de résolution.

Chapitre 3

Méthodes incrémentales pour la décision multicritère sur domaine combinatoire

Ce chapitre s'appuie sur nos travaux publiés dans [Benabbou and Perny, 2015a,b,c] s'articulant autour de la problématique de l'élicitation des préférences pour la décision multicritère sur domaine combinatoire. Dans ce contexte, pour parvenir à détecter efficacement la meilleure alternative pour le décideur, nous proposons une nouvelle approche consistant à mêler résolution du problème d'optimisation et élicitation incrémentale des préférences. Plus précisément, nous commençons par concevoir des algorithmes permettant de déterminer l'ensemble des solutions potentiellement optimales en présence de préférences imprécisément connues. Puis, nous examinons quand poser des questions durant l'exécution de ces algorithmes pour pouvoir identifier une solution nécessairement optimale à la fin de la résolution. Pour limiter le nombre de questions posées, nous adoptons l'approche d'élicitation incrémentale fondée sur la notion de regret pour engendrer des questions informatives durant la recherche. Dans ce chapitre, nous étudions le potentiel de cette nouvelle approche, combinant recherche et élicitation, dans le cadre de deux problèmes d'optimisation combinatoire particuliers : la recherche dans un graphe d'états multi-objectifs et le problème d'arbres couvrants multi-objectifs. Pour ces problèmes, nous proposons des algorithmes reposant respectivement sur la programmation dynamique et sur le principe glouton.

Dans ce chapitre, nous considérons une situation de décision multicritère où l'ensemble $\mathcal{X}$ des alternatives possibles est défini implicitement comme étant l'ensemble des solutions réalisables d'un problème d'optimisation combinatoire multi-objectifs. Plus précisément, nous allons étudier la problématique de recherche dans un graphe d'états multicritère avant de nous consacrer aux problèmes d'arbres couvrants multi-objectifs. L'élicitation des préférences sur domaine combinatoire est une problématique à l'origine de contributions récentes dans de nombreux contextes, comme par exemple dans les problèmes de satisfaction de contraintes (e.g., [Boutilier et al., 2006, Gelain et al., 2010]), les espaces multi-attributs (e.g., [Gonzales and Perny, 2004, Braziunas and Boutilier, 2007, Koriche and Zanuttini, 2010]), les problèmes de planification (e.g., [Regan and Boutilier, 2011, Weng and Zanuttini, 2013, Gilbert et al., 2015]), les problèmes de mariages stables (e.g., [Drummond and Boutilier, 2014]) ou encore les problèmes d'optimisation multi-objectifs [Vanderpooten and Vincke, 1997, Korhonen, 2005, Kaddani et al., 2017]. Notre objectif est de proposer une nouvelle approche, mêlant élicitation et résolution, permettant de réduire significativement la taille de l'espace de recherche tout en limitant le nombre de questions posées au décideur. À titre préliminaire, nous nous intéressons aussi à la problématique de détermination des solutions potentiellement optimales en présence de préférences imprécisément connues. Cette problématique a été considérée dans de nombreux contextes, comme par exemple en décision multi-agents (e.g., [Konczak and Lang, 2005, Xia and Conitzer, 2011, Roijers et al., 2014, Aziz et al., 2015b]) ou en satisfaction de contraintes (e.g., [Marinescu et al., 2013, Wilson et al., 2015]).

3.1 Recherche dans un graphe d'états multicritère

La recherche fondée sur les préférences est une problématique très étudiée dans le domaine de l'intelligence artificielle avec des applications diverses, notamment en satisfaction de contraintes, planification, allocation de ressources et commerce électronique (e.g., [Boutilier et al., 2004, Perny and Spanjaard, 2005, Brafman and Domshlak, 2009, Domshlak et al., 2011, Galand et al., 2013]). Bien que la plupart des contributions algorithmiques se concentrent sur la détermination efficace des solutions préférées pour un modèle de préférences donné, d'autres travaux se sont penchés sur la mise en œuvre de stratégies d'élicitation permettant de représenter au mieux les préférences du décideur. Dans cette sous-section, il s'agit pour nous de considérer ces deux aspects simultanément pour limiter à la fois les temps de résolution et le nombre de questions nécessaires à la prise de décision. Plus précisément, nous étudions ici le potentiel de l'élicitation incrémentale (e.g., [White III et al., 1984, Wang and Boutilier, 2003, Braziunas and Boutilier, 2007]) combinée à la recherche dans un graphe d'états multicritère (e.g., [Stewart and White III, 1991, Mandow and Pérez de la Cruz, 2005b]).

Dans cette sous-section, nous considérons donc un graphe d'états G doté de plusieurs fonctions de coût (ou critères d'évaluation) qu'il convient de minimiser (e.g., temps, distance, énergie, risque). Chaque chemin de ce graphe est ainsi associé à un vecteur $(x_1, \ldots, x_q) \in \mathbb{R}^q_+$ où x_j représente sa performance sur le critère $j \in \mathcal{Q} = \{1, \ldots, q\}$. Dans ce graphe, les solutions réalisables sont tous les chemins partant du *nœud source* et arrivant à un *nœud but* du problème. Nous supposons ici que les préférences du décideur

sur les chemins découlent de ses préférences sur les vecteurs coût, ces dernières étant représentables par une fonction d'agrégation $f_\omega : \mathbb{R}_+^q \to \mathbb{R}_+$ de paramètre ω définissant le coût global (ou désutilité) de tout vecteur $x \in \mathbb{R}_+^q$. Dans ce contexte, il s'agit de déterminer une solution réalisable minimisant la fonction f_ω modélisant les préférences subjectives du décideur.

Lorsque la fonction f_ω est imprécisément connue, il convient de mettre en place des stratégies d'élicitation de ω pour pouvoir détecter une solution optimale pour le décideur. Sur domaine combinatoire, l'approche classique consiste à balayer l'espace des paramètres, en engendrant à plusieurs reprises une instance possible de ω et une solution optimale associée, tant que les solutions qui ont été engendrées ne conviennent pas au décideur (e.g., [Zionts and Wallenius, 1976, Vanderpooten and Vincke, 1997]). Dans le cadre de la problématique de recherche dans un graphe d'état, la génération d'une solution optimale peut être réalisée à chaque étape de manière efficace lorsque la fonction f_ω est linéaire. En effet, il suffit d'appliquer l'algorithme A* [Hart et al., 1968] sur le graphe obtenu en agrégeant les coûts avec cette fonction. Dans la littérature, on trouve aussi de nombreux algorithmes de résolution pour répondre à cette problématique avec des fonctions d'agrégation plus complexes (voir par exemple [Dasgupta et al., 1995] pour une fonction d'utilité multi-attribut, [Galand and Perny, 2006] pour les normes de Tchebycheff pondérées, et [Galand and Perny, 2007, Galand et al., 2013] pour les intégrales de Choquet).

Dans l'optique d'éliciter les paramètres ω, nous considérons ici une autre manière de procéder, consistant essentiellement à intégrer l'élicitation à la recherche. Plus précisément, il s'agit de produire des questions à des moments clés de la résolution, de manière à diminuer la taille de l'espace de recherche tout en garantissant de détecter une solution nécessairement optimale à la fin de l'exécution. Pour présenter notre approche, nous allons commencer par introduire formellement la problématique de recherche des solutions préférées dans un graphe d'états. Ensuite, nous proposerons un algorithme efficace pour la recherche des solutions potentiellement optimales lorsque les préférences du décideur sont représentables par une somme pondérée f_ω imprécisément connue (une solution est dite potentiellement optimale si et seulement si celle-ci minimise f_ω pour une instance possible de ω). Puis, nous proposerons des stratégies de génération de questions permettant de réduire l'imprécision sur les paramètres du modèle en cours de résolution de manière à détecter rapidement une solution préférée. Enfin, nous étudierons un modèle décisionnel plus général que la somme pondérée, à savoir l'intégrale de Choquet.

3.1.1 Introduction et formalisation du problème

Soit $G = (N, A)$ un graphe d'états défini implicitement, où N est un ensemble fini de nœuds représentant les états du graphe et A est un ensemble d'arcs modélisant les transitions possibles entre états. Plus formellement, nous avons $A = \{(n, n') : n \in V, n' \in \Pi(n)\}$ où $\Pi(n) \subseteq N$ est l'ensemble des successeurs de l'état n. Un chemin du nœud n au nœud n' dans le graphe G est caractérisé par une liste de nœuds de type $\langle n_1, \ldots, n_m \rangle$, où $n_1 = n$, $n_m = n'$ et $(n_k, n_{k+1}) \in A$ pour tout $k \in \{1, \ldots, m-1\}$. L'ensemble de tous les chemins partant du nœud n et arrivant au nœud n' sera noté $P(n, n')$ dans la suite. Parmi tous les nœuds du graphe, nous distinguons en particulier le *nœud source*, noté s, ainsi que

l'ensemble des *nœuds buts*, noté Γ. Les chemins solutions sont alors tous ceux qui partent du nœud source s et qui arrivent à un nœud but $\gamma \in \Gamma$; l'ensemble de ces chemins sera noté $P(s, \Gamma)$ ci-après.

Par ailleurs, chaque arc $a \in A$ est évalué sur un ensemble fini de critères $g_j : A \to \mathbb{R}_+$, $j \in \mathcal{Q} = \{1, \ldots, q\}$, représentant des coûts à minimiser. Le graphe G est ainsi doté d'une fonction d'évaluation $g : A \to \mathbb{R}_+^q$ qui associe à chaque arc $a \in A$ le vecteur coût $g(a) = (g_1(a), \ldots, g_q(a))$. Dans ce graphe, le vecteur coût d'un chemin p, noté $g(p)$, est égal à la somme des vecteurs coût des arcs qui le constituent. Ainsi, nous nous intéressons à l'ensemble $\mathcal{X} = \{g(p) : p \in P(s, \Gamma)\}$ représentant l'image de tous les chemins solutions dans l'espace des critères. Plus précisément, il s'agit pour nous de déterminer le vecteur coût de cet ensemble que le décideur préfère.

La recherche de solutions préférées en optimisation multi-objectifs est souvent fondée sur l'exploration de l'ensemble des solutions non Pareto-dominées :

$$\mathrm{ND}(\mathcal{X}) = \{x \in \mathcal{X} : \forall y \in \mathcal{X}, \neg(y \prec_P x)\}$$

où $\neg$ est l'opérateur de négation logique et $y \prec_P x$ si et seulement si $y_j \leq x_j$ pour tout $j \in \mathcal{Q}$ et $y_j < x_j$ pour au moins un critère $j \in \mathcal{Q}$. En effet, la dominance de Pareto permet d'éliminer du problème des alternatives qui sont indéniablement sous-optimales, puisque celles-ci sont par définition battues par une autre alternative sur tous les critères à la fois. Par ailleurs, la dominance de Pareto présente l'avantage pratique de respecter ce que l'on appelle le principe de Bellman : toute sous-solution d'une solution Pareto-optimale est elle aussi Pareto-optimale sur le sous-problème correspondant. Pour la recherche dans les graphes d'états, cela revient à dire que tout sous-chemin d'un chemin Pareto-optimal est Pareto-optimal parmi ceux ayant la même destination. Cette propriété a conduit les chercheurs à proposer des extensions multicritères d'algorithmes constructifs performants dédiés à la résolution de problèmes mono-critères. En particulier, l'algorithme MOA* (e.g., [Stewart and White III, 1991, Mandow and Pérez de la Cruz, 2005b]) est une extension multicritère de l'algorithme A* (e.g., [Hart et al., 1968]) permettant de déterminer l'ensemble $\mathrm{ND}(\mathcal{X})$ et de retourner un chemin solution par élément de cet ensemble. Nous présentons ci-après les grandes lignes de ces deux algorithmes.

Algorithme A* [Hart et al., 1968]

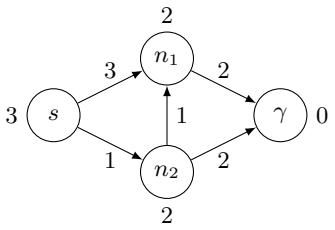
L'algorithme A* est un algorithme de programmation dynamique permettant de déterminer efficacement un chemin optimal pour une fonction coût scalaire en utilisant une fonction heuristique. La fonction heuristique permet d'estimer le coût du meilleur chemin partant de tout nœud $n \in N$ du graphe et arrivant à un des nœuds buts du problème. Pour des graphes de grande taille, la fonction heuristique joue un rôle crucial car celle-ci est utilisée pour choisir de manière informée le nœud qu'il convient de développer pour limiter l'exploration du graphe. L'algorithme A* procède par étiquetage des nœuds. Plus précisément, à tout moment de l'algorithme, chaque nœud $n \in N \backslash \{s\}$ découvert est étiqueté avec :

- le coût du meilleur chemin de s à n trouvé jusque là, noté g_n,
- et le nœud précédant n sur ce chemin, appelé *nœud père*.

Initialement, seul le nœud source s est découvert et nous avons $g_s = 0$. La recherche d'un chemin solution minimisant la fonction coût se fait alors en développant les nœuds du graphe de manière itérative. Lorsqu'un nœud n est développé, tous ses successeurs n' sont mis à jour en comparant le coût $g_{n'}$ avec la somme du coût g_n et du coût de l'arc (n, n'). Plus précisément, l'étiquetage du nœud n' ne change que si l'inégalité $g_{n'} > g_n + g(n, n')$ est vérifiée. Dans ce cas, la seconde valeur remplace le coût $g_{n'}$ et n devient le nouveau nœud père de n'. En effet, nous venons de découvrir un chemin de s à n' (passant par n) dont le coût est strictement inférieur à tous ceux que nous avions trouvés jusque là. Si le nœud n' a été découvert en développant le nœud n, alors sont coût $g_{n'}$ est tout simplement initialisé à la valeur $g_n + g(n, n')$ et n est son nœud père initial.

Le choix du prochain nœud à développer se fait parmi les nœuds découverts non encore développés, suivant une fonction f de la forme $f(n) = g_n + h(n)$, où h est une fonction heuristique associant à tout nœud $n \in N$ une estimation du coût du meilleur chemin partant de n et arrivant à un nœud but. Le prochain nœud à développer est celui qui minimise la fonction d'évaluation f. L'algorithme s'arrête lorsque le prochain nœud à développer s'avère être un nœud but $\gamma \in \Gamma$. Le chemin retourné par l'algorithme est ensuite construit à partir des nœuds pères, en commençant par celui du nœud γ, jusqu'à arriver au nœud source s. À titre illustratif, nous détaillons ci-après l'exécution de cet algorithme sur un graphe de petite taille :

Exemple 21. *Considérons le graphe $G = (N, A)$, avec $N = \{s, n_1, n_2, \gamma\}$, représenté ci-dessous :*



Sur cette figure, le coût $g(a)$ est indiqué au milieu de chaque arc $a \in A$ et la valeur $h(n)$ est renseignée à côté de chaque nœud $n \in N$. Notons O l'ensemble des nœuds découverts qui n'ont pas encore été développés ; initialement, nous avons $O = \{s\}$ avec $g_s = 0$. Durant la première itération, le nœud s est retiré de l'ensemble O pour être développé. Comme nous avons $\Pi(s) = \{n_1, n_2\}$ et $\Pi(s) \cap O = \emptyset$, alors tous les nœuds $n \in \Pi(s)$ sont insérés dans O et le nœud s devient leur nœud père. Par ailleurs, l'algorithme étiquette ces nœuds selon la formule $g_n = g_s + g(s, n)$, ce qui nous donne $g_{n_1} = 3$ et $g_{n_2} = 1$. Pour la deuxième itération, comme $f(n_2) = 1 + 2 = 3 < f(n_1) = 3 + 1 = 4$, alors l'algorithme retire le nœud n_2 de l'ensemble O afin de le développer. À cette itération, les nœuds à mettre à jour sont ceux de l'ensemble $\Pi(n_2) = \{n_1, \gamma\}$. Comme $\gamma \notin O$, alors γ est inséré dans O, le nœud n_2 devient son nœud père et l'algorithme étiquette γ avec $g_\gamma = g_{n_2} + g(n_2, \gamma) = 1 + 2 = 3$. Pour le nœud $n_1 \in O$, nous devons comparer g_{n_1} avec $g_{n_2} + g(n_2, n_1)$. Comme $g_{n_1} = 3 > g_{n_2} + g(n_2, n_1) = 1 + 1 = 2$, alors l'algorithme réalise la mise à jour suivante : $g_{n_1} = 2$ et n_2 devient le nouveau nœud père de n_1. À la troisième itération, c'est le nœud γ qui est développé car nous avons $f(\gamma) = 3 + 0 = 3 < f(n_1) = 2 + 2 = 4$. Puisque γ est nœud but, l'algorithme s'arrête immédiatement en retournant le chemin $\langle s, n_2, \gamma \rangle$.

La validité de l'algorithme A* repose essentiellement sur la fonction heuristique h utilisée. Plus précisément, l'algorithme est valide si et seulement si h est *admissible*, ce qui signifie que, pour tout nœud n du graphe, la valeur $h(n)$ doit être inférieure ou égale au coût du meilleur chemin allant de n à un nœud but. En effet, avec une heuristique admissible, lorsqu'un nœud but est développé, nous savons que son coût est inférieur à l'estimation heuristique de tous les autres nœuds, qui est elle-même inférieure au coût des meilleurs chemins issus de ces nœuds. Soulignons que le célèbre algorithme de Dijkstra [Dijkstra, 1959], conçus pour l'optimisation mono-critère dans les graphes définis explicitement, se retrouve en prenant tout simplement la fonction nulle comme heuristique.

Algorithme MOA* [Mandow and Pérez de la Cruz, 2005b]

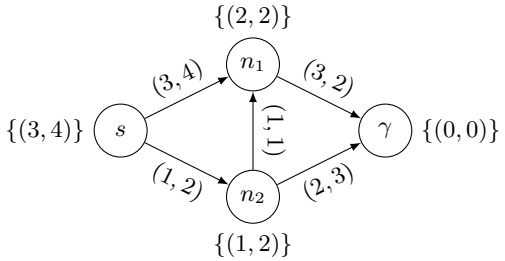
L'algorithme MOA* a été introduit par Stewart and White III [1991] pour étendre l'algorithme A* aux graphes avec une évaluation multicritère. Il s'agit de déterminer l'ensemble $\mathrm{ND}(\mathcal{X})$ des vecteurs coût Pareto-optimaux et un chemin par élément de cet ensemble. Nous présentons ici la version proposée dans l'article [Mandow and Pérez de la Cruz, 2005b], aussi appelée NAMOA* pour *New Approach to Multiobjective A**. Cette version s'est en effet révélée beaucoup plus efficace en pratique que l'algorithme d'origine, en passant simplement du développement de nœuds au développement d'étiquettes.

Dans le cas multi-objectifs, plusieurs chemins de même provenance et de même destination peuvent être optimaux au sens de la dominance de Pareto. De ce fait, l'algorithme MOA* explore le graphe en étendant des étiquettes correspondant à des chemins au lieu de développer des nœuds. Plus précisément, chaque nœud du graphe peut maintenant se voir attribuer plusieurs étiquettes, une étiquette ℓ par chemin découvert p_ℓ arrivant à ce nœud. Chaque étiquette ℓ est sous la forme $\ell = [n_\ell, p_\ell, g_\ell]$ où n_ℓ est le nœud d'arrivée du chemin p_ℓ et g_ℓ est le vecteur coût associé à ce chemin. Quand une étiquette ℓ est développée, chaque successeur n' du nœud n_ℓ reçoit une nouvelle étiquette ℓ' correspondant au chemin obtenu en ajoutant l'arc (n_ℓ, n') au chemin p_ℓ. Chaque nœud successeur n' est ensuite mis à jour en supprimant toutes les étiquettes arrivant à ce nœud dont le vecteur coût est Pareto-dominé. Rappelons que ces suppressions sont bien autorisées car la dominance de Pareto vérifie le principe de Bellman (tout chemin solution Pareto-optimal est composé de chemins Pareto-optimaux).

La prochaine étiquette à développer est sélectionnée parmi les étiquettes non développées, le choix se faisant sur la base de leurs évaluations heuristiques. En effet, chaque nœud $n \in N$ est associé à un ensemble de vecteurs heuristiques $H(n)$ estimant l'ensemble des vecteurs coût $\{g(p) : p \in P(n, \Gamma)\}$. Ainsi, chaque étiquette ℓ admet un ensemble $F(\ell)$ de vecteurs coût heuristiques qui est défini par $F(\ell) = \{g_\ell + h : h \in H(n_\ell)\}$. La prochaine étiquette à développer est alors choisie parmi celles ayant au moins un vecteur heuristique qui n'est Pareto-dominé par aucun autre vecteur heuristique (associé à une étiquette non développée). L'idée est de se concentrer sur les chemins semblant mener à une solution Pareto-optimale. Par ailleurs, l'algorithme MOA* se distingue de son homologue A* en utilisant les chemins solutions découverts pour supprimer des étiquettes non développées. Plus précisément, si tous les éléments de $F(\ell)$ sont dominés au sens de Pareto par un chemin solution découvert, alors l'étiquette ℓ est supprimée par l'algorithme MOA*. L'algorithme se termine quand toutes les étiquettes engendrées ont été développées

ou supprimées. Pour donner une illustration de cet algorithme de recherche, nous déroulons ci-après son exécution sur exemple simple :

Exemple 22. *Considérons le graphe bicritère $G = (N, A)$ suivant :*



Sur cette figure, le vecteur coût $g(a)$ est renseigné au milieu de chaque arc $a \in A$ et l'ensemble $H(n)$ est indiqué à côté de chaque nœud $n \in N$. Notons O l'ensemble des étiquettes engendrées mais pas encore développées ; initialement, nous avons $O = \{\ell_s\}$ où $\ell_s = [s, \langle s \rangle, 0]$. Durant la première itération, l'étiquette ℓ_s est retirée de l'ensemble O pour être développée. Comme nous avons $\Pi(s) = \{n_1, n_2\}$, alors les deux étiquettes suivantes sont insérées dans l'ensemble O : $\ell_1 = [n_1, \langle s, n_1 \rangle, (3, 4)]$ et $\ell_2 = [n_2, \langle s, n_2 \rangle, (1, 2)]$. Ces dernières sont donc les deux étiquettes candidates au prochain développement. Notons que nous avons $F(\ell_1) = \{(5, 6)\}$, $F(\ell_2) = \{(2, 4)\}$ et $(2, 4) \prec_P (5, 6)$. Par conséquent, durant la deuxième itération, l'algorithme choisit de retirer l'étiquette ℓ_2 de l'ensemble O pour la développer. Comme $\Pi(n_2) = \{n_1, \gamma\}$, alors ce développement engendre les étiquettes suivantes : $\ell'_1 = [n_1, \langle s, n_2, n_1 \rangle, (2, 3)]$ et $\ell_\gamma = [\gamma, \langle s, n_2, \gamma \rangle, (3, 5)]$. Ces étiquettes sont ensuite insérées dans l'ensemble O. Néanmoins, comme ℓ_1 et ℓ'_1 correspondent à deux chemins arrivant au même nœud, alors l'algorithme compare leur vecteur coût pour garder uniquement les étiquettes non dominées : l'étiquette ℓ_1 est ici supprimée puisque nous avons $g_{\ell'_1} \prec_P g_{\ell_1}$. Par ailleurs, comme l'étiquette ℓ_γ correspond à un chemin solution, alors l'algorithme doit supprimer de l'ensemble O toute étiquette ℓ vérifiant $g_{\ell_\gamma} \prec_P f$ pour tout $f \in F(\ell)$. Par conséquent, l'étiquette ℓ'_1 est à son tour supprimée de cet ensemble car nous avons $F(\ell') = \{(4, 5)\}$ et $g_{\ell_\gamma} = (3, 5) \prec_P (4, 5)$. En résumé, l'ensemble O ne contient plus que l'étiquette ℓ_γ à la fin de la deuxième itération. À la troisième itération, c'est donc cette étiquette qui est retirée de l'ensemble O pour être développée. Comme $\Pi(\gamma) = \emptyset$, alors l'algorithme n'engendre aucune nouvelle étiquette à cette itération. Puis, l'algorithme se termine car l'ensemble O est à présent vide.

Cet algorithme est valide si et seulement si H est *admissible*, autrement dit si et seulement si, pour tout nœud $n \in N$ et pour tout chemin $p \in P(n, \Gamma)$, il existe un vecteur heuristique $h \in H(n)$ tel que $h \precsim_P y$ (i.e. $h_j \leq y_j$ pour tout $j \in \mathcal{Q}$). Cette propriété permet d'assurer que toute étiquette supprimée par comparaison avec des chemins solutions découverts ne peut conduire à un chemin solution Pareto-optimal. En effet, tous les chemins solutions issus de cette étiquette sont dominés au sens de Pareto par au moins un de ses vecteurs heuristiques (car H est admissible), et ces vecteurs sont eux-mêmes dominés par un chemin solution découvert (car cette étiquette est supprimée par l'algorithme).

La nécessité de résoudre des instances de plus en plus grandes de ce problème a suscité des travaux permettant d'améliorer la recherche effectuée par l'algorithme MOA*. Par exemple, favoriser une explora-

tion en profondeur du graphe de recherche semble en pratique améliorer les performances de l'algorithme, en lui permettant de détecter plus rapidement des chemins solutions (e.g., [Harikumar and Kumar, 1996, Mandow and Pérez de la Cruz, 2005b, Coego et al., 2009, 2013]). Par ailleurs, il est important de souligner que l'efficacité de ces algorithmes dépend fortement des fonctions heuristiques utilisées pour diriger la recherche et éliminer des chemins par comparaison avec des chemins solutions. En particulier, il a été démontré que l'utilisation d'une fonction heuristique monotone (i.e. $h(n) \precsim_P g(n,n') + h(n')$ pour tout $n' \in \Pi(n)$) permet de réduire le nombre d'étiquettes développées (e.g., [Mandow and Pérez de la Cruz, 2005a]). De manière générale, l'élaboration de bonnes heuristiques multicritères est une problématique relativement complexe en soi, car il s'agit de recueillir des informations suffisamment précises sur les performances des solutions optimales, solutions que nous ne connaissons pas avant de commencer la résolution (voir par exemple [Dasgupta et al., 1995] pour des travaux sur cette problématique).

Comme nous l'avons déjà mentionné, nous supposons dans cette section que les préférences du décideur sont représentables par une fonction d'agrégation $f_\omega : \mathbb{R}_+^q \to \mathbb{R}_+$ dont les paramètres ω sont imprécisément connus. Cette fonction permet de mesurer le coût global de tout chemin du graphe du point de vue du décideur. Dans ce cadre, un vecteur coût $x \in \mathcal{X}$ est dit *potentiellement optimal* si et seulement si celui-ci minimise la fonction f_ω pour une instance admissible des paramètres ω. Dans un premier temps, nous supposerons que les préférences du décideur peuvent être modélisées par une somme pondérée, et nous étudierons la problématique de détermination des vecteurs solutions potentiellement optimaux dans ce contexte. En particulier, nous proposerons un algorithme de résolution utilisant MOA* comme structure de base. Puis, nous montrerons comment combiner cet algorithme avec l'élicitation incrémentale pour détecter efficacement un vecteur solution optimal. Enfin, nous nous intéresserons à un modèle décisionnel plus général, à savoir l'intégrale de Choquet.

3.1.2 Calcul des solutions potentiellement optimales pour la somme pondérée

Dans cette sous-section, nous supposons que les préférences du décideur sont représentables par une somme pondérée f_ω, le jeu de poids ω étant imprécisément connu (cf. définition 11). Notons $\mathcal{P}$ l'ensemble des données que nous avons pu collecter sur les préférences du décideur. Pour simplifier la présentation, nous supposons que $\mathcal{P}$ est composé de paires (a,b) représentant la préférence "le vecteur coût a est meilleur que le vecteur coût b". Notons à présent Ω l'ensemble des jeux de poids ω compatibles avec l'information $\mathcal{P}$. Si aucune information n'est disponible, alors Ω est tout simplement constitué de tous les jeux de poids strictement positifs, i.e. $\Omega = \{\omega \in \text{int}(\mathbb{R}_+^q) : \sum_{j=1}^q \omega_j = 1\}$, où int représente l'intérieur du cône. Par ailleurs, puisque f_ω est linéaire en ω, alors toute préférence de type "a est meilleur que b" induit la contrainte linéaire $f_\omega(a) \leq f_\omega(b)$ sur l'ensemble de jeux de poids Ω. De ce fait, l'ensemble Ω est ici un polyèdre convexe. Dans ce contexte, nous nous intéressons à la détermination de l'ensemble $\text{PO}_\Omega(\mathcal{X})$ des vecteurs coût potentiellement optimaux, formellement défini comme suit :

Définition 46. $\text{PO}_\Omega(\mathcal{X}) = \bigcup_{\omega \in \Omega} \arg\min_{x \in \mathcal{X}} f_\omega(x)$

Par définition, $\text{PO}_\Omega(\mathcal{X})$ est l'ensemble des vecteurs coût (associés à un chemin solution) minimisant la

fonction f_ω pour au moins un jeu de poids $\omega \in \Omega$. Pour déterminer l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$, nous présentons tout d'abord une méthode en deux phases, puis nous proposerons un algorithme plus efficace fondé sur la programmation dynamique.

i) Une méthode en deux phases

Dans la section 1.1.2, nous avons vu que la somme pondérée est monotone avec la dominance de Pareto ; en particulier, nous avons $f_\omega(x) < f_\omega(y)$ pour tout $x, y \in \mathcal{X}$ tels que $x \prec_P y$. Cette propriété implique en particulier l'égalité suivante : $\mathrm{PO}_\Omega(\mathcal{X}) = \mathrm{PO}_\Omega(\mathrm{ND}(\mathcal{X}))$. Par conséquent, l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ peut être obtenu par une méthode en deux phases, consistant tout d'abord à utiliser MOA* pour calculer $\mathrm{ND}(\mathcal{X})$, puis à déterminer $\mathrm{PO}_\Omega(\mathcal{X})$ à partir de l'ensemble $\mathrm{ND}(\mathcal{X})$. Nous proposons ici un algorithme réalisant cette seconde phase de manière efficace. En d'autres termes, il s'agit pour nous de proposer un algorithme permettant de déterminer efficacement l'ensemble $\mathrm{PO}_\Omega(X)$ des solutions potentiellement optimales pour un ensemble fini de solutions $X = \{x_1, \ldots, x_m\}$ donné en extension. L'algorithme que nous proposons est fondé sur la relation de dominance suivante :

Définition 47 (Ω-dominance)**.** *Pour tout* $y \in \mathbb{R}^q_+$ *et tout* $Z \subseteq \mathbb{R}^q_+$ *:* $Z \prec_\Omega y \Leftrightarrow \forall \omega \in \Omega, \exists z \in Z, f_\omega(z) < f_\omega(y)$.

Cette relation est en effet très utile pour calculer $\mathrm{PO}_\Omega(X)$ car celle-ci vérifie la propriété suivante :

Proposition 17. *Pour tout* $y \in X$ *:* $y \in \mathrm{PO}_\Omega(X) \Leftrightarrow \forall Z \subseteq X, \neg(Z \prec_\Omega y)$.

Démonstration. ($\Rightarrow$) Soit $y \in \mathrm{PO}_\Omega(X)$. Supposons qu'il existe $Z \subseteq X$ tel que $Z \prec_\Omega y$. Dans ce cas, pour tout $\omega \in \Omega$, il existe $z \in Z$ tel que $f_\omega(z) < f_\omega(y)$ (cf. définition 47). Comme en plus nous avons $Z \subseteq X$ par hypothèse, alors nous en déduisons que y n'est pas un élément de $\arg\min_{x \in X} f_\omega(x)$ pour tout $\omega \in \Omega$; ceci contredit l'hypothèse $y \in \mathrm{PO}_\Omega(X)$.

($\Leftarrow$) Soit $y \in X$ tel que nous avons $\neg(Z \prec_\Omega y)$ pour tout $Z \subseteq X$. Notons que nous avons $\neg(X \prec_\Omega y)$ en particulier. En utilisant la définition 47, nous en déduisons qu'il existe $\omega \in \Omega$ tel que $f_\omega(z) \geq f_\omega(y)$ pour tout $z \in X$. Ainsi, nous avons forcément $y \in \mathrm{PO}_\Omega(X)$ dans ce cas là. □

Ce résultat nous a conduit à proposer l'algorithme 3 (appelé Ω-Filter ci-après) pour calculer $\mathrm{PO}_\Omega(X)$.

Algorithm 3: Ω-Filter(X)

Input: Ω ; $X = \{x^1, \ldots, x^m\}$; σ : permutation de $\{1, \ldots, m\}$

```
1 X_0^σ ← X
2 for i = 1...m do
3     if X_{i-1}^σ ≺_Ω x^{σ(i)} then
4         X_i^σ ← X_{i-1}^σ \ {x^{σ(i)}}
5     else
6         X_i^σ ← X_{i-1}^σ
7     end
8 end
9 return X_m^σ
```

Avant de prouver la validité de notre algorithme, nous allons montrer que celui-ci s'exécute en temps polynomial. Pour déterminer l'ensemble $\mathrm{PO}_\Omega(X)$, Ω-Filter construit une séquence d'ensembles emboîtés X_i^σ, $i = 1 \dots m$, en testant itérativement si $X_{i-1}^\sigma \prec_\Omega x^{\sigma(i)}$, où σ est une permutation donnée de l'ensemble $\{1, \dots, m\}$; ainsi, Ω-Filter se résume à la réalisation de m tests de Ω-dominance. Par ailleurs, rappelons que Ω est décrit par un ensemble raisonnablement faible de contraintes linéaires (au plus une contrainte par élément de $\mathcal{P}$). Par conséquent, chaque test $X_{i-1}^\sigma \prec_\Omega x^{\sigma(i)}$ peut être effectué en résolvant le programme linéaire suivant :

$$\max_{\omega \in \Omega} \min_{x \in X_{i-1}^\sigma} \left\{ f_\omega(x) - f_\omega(x^{\sigma(i)}) \right\}$$

Ceci est une simple conséquence de la propriété suivante :

Proposition 18. *Pour tout $y \in \mathbb{R}_+^q$ et tout $Z \subseteq \mathbb{R}_+^q$:*

$$Z \prec_\Omega y \;\Leftrightarrow\; \left[\max_{\omega \in \Omega} \min_{z \in Z} \left\{ f_\omega(z) - f_\omega(y) \right\} < 0 \right]$$

Démonstration. ($\Rightarrow$) Soient $y \in \mathbb{R}_+^q$ et $Z \subseteq \mathbb{R}_+^q$ tels que $Z \prec_\Omega y$. Pour tout $\omega \in \Omega$, il existe $z \in Z$ tels que $f_\omega(z) < f_\omega(y)$ (par définition 47). Par conséquent, nous avons $\min_{z \in Z}\{f_\omega(z) - f_\omega(y)\} < 0$ pour tout $\omega \in \Omega$ et donc nous avons en particulier $\max_{\omega \in \Omega} \min_{z \in Z}\{f_\omega(z) - f_\omega(y)\} < 0$.

($\Leftarrow$) Soient $y \in \mathbb{R}_+^q$ et $Z \subseteq \mathbb{R}_+^q$ tels que $\max_{\omega \in \Omega} \min_{z \in Z}\{f_\omega(z) - f_\omega(y)\} < 0$. Pour tout $\omega \in \Omega$, nous avons donc $\min_{z \in Z}\{f_\omega(z) - f_\omega(y)\} < 0$. Par conséquent, pour tout $\omega \in \Omega$, il existe $z \in Z$ tel que $f_\omega(z) < f_\omega(y)$. D'où $Z \prec_\Omega y$ (par définition 47). □

Finalement, nous venons de montrer que Ω-Filter doit résoudre m programmes linéaires de taille bornée, ce qui peut être réalisé en temps polynomial par programmation linéaire. Montrons à présent que notre algorithme est valide. Pour ce faire, nous allons tout d'abord prouver que le résultat de celui-ci ne dépend pas de la permutation σ utilisée (autrement dit, la numérotation des solutions x^i, $i \in \{1, \dots, m\}$, n'a pas d'importance).

Proposition 19. *Pour toutes permutations σ_1 et σ_2, nous avons $X_m^{\sigma_1} = X_m^{\sigma_2}$, où $X_m^{\sigma_1}$ (resp. $X_m^{\sigma_2}$) est l'ensemble retourné par l'algorithme Ω-Filter avec σ_1 (resp. σ_2).*

Démonstration. Soient σ_1 et σ_2 deux permutations de $\{1, \dots, m\}$. Montrons par l'absurde que nous avons $X_m^{\sigma_1} \subseteq X_m^{\sigma_2}$. Soit $x^i \in X_m^{\sigma_1}$. Supposons $x^i \notin X_m^{\sigma_2}$. Soit $k \in \{1, \dots, m\}$ l'itération telle que $i = \sigma_1(k)$. Puisque $x^i \in X_m^{\sigma_1}$, alors nous avons nécessairement $\neg(X_{k-1}^{\sigma_1} \prec_\Omega x^i)$; autrement dit, il existe forcément $\omega' \in \Omega$ tel que $f_{\omega'}(x) \geq f_{\omega'}(x^i)$ pour tout $x \in X_{k-1}^{\sigma_1}$ (cf. définition 47). Notons k' l'itération telle que $i = \sigma_2(k')$. Puisque $x^i \notin X_m^{\sigma_2}$, nous avons cette fois-ci $X_{k'-1}^{\sigma_2} \prec_\Omega x^i$; autrement dit, pour tout $\omega \in \Omega$, il existe $x \in X_{k'-1}^{\sigma_2}$ tel que $f_\omega(x) < f_\omega(x^i)$ (cf. définition 47). Par conséquent, l'ensemble $X' = \{x \in X : f_{\omega'}(x) < f_{\omega'}(x^i)\}$ n'est pas vide car nous avons $X_{k'-1}^{\sigma_2} \subseteq X$ par construction. En utilisant la définition de ω', nous en déduisons $X' \cap X_{k-1}^{\sigma_1} = \emptyset$. Puisque $X_{k-1}^{\sigma_1} \supseteq X_m^{\sigma_1}$ par construction, alors nous avons $X' \cap X_m^{\sigma_1} = \emptyset$. Ainsi, pour tout $l \in \{1, \dots, m\}$ tel que $x^{\sigma_1(l)} \in X'$, nous avons nécessairement $x^{\sigma_1(l)} \in X_{l-1}^{\sigma_1}$ et $x^{\sigma_1(l)} \notin X_l^{\sigma_1}$. Par conséquent, nous avons $X' \cap X_{L-1}^{\sigma_1} = \{x^{\sigma_1(L)}\}$, où L représente la plus

grande itération l telle que $x^{\sigma_1(l)} \in X'$. Puis, comme nous avons $x^{\sigma_1(L)} \notin X_L^{\sigma_1}$, alors $X_{L-1}^{\sigma_1} \prec_\Omega x^{\sigma_1(L)}$ est nécessairement vrai. Il existe donc forcément $x' \in X_{L-1}^{\sigma_1}$ tel que $f_{\omega'}(x') < f_{\omega'}(x^{\sigma_1(L)})$ (cf. définition 47). Puisque $X' \cap X_{L-1}^{\sigma_1} = \{x^{\sigma_1(L)}\}$, nous savons aussi que $x' \notin X'$. Finalement, comme $x^{\sigma_1(L)} \in X'$ par définition, alors nous avons $f_{\omega'}(x') < f_{\omega'}(x^{\sigma_1(L)}) < f_{\omega'}(x^i)$, ce qui contredit le fait que x' n'est pas dans X'. Par conséquent, nous avons forcément $X_m^{\sigma_1} \subseteq X_m^{\sigma_2}$. Comme σ_1 et σ_2 sont des permutations quelconques, alors nous pouvons montrer de manière similaire que nous avons aussi $X_m^{\sigma_2} \subseteq X_m^{\sigma_1}$, ce qui permet de conclure la preuve. □

Ainsi, l'algorithme Ω-Filter retourne exactement le même sous-ensemble de X pour toute permutation σ de $\{1, \ldots, m\}$. Ce résultat est utilisé ci-dessous pour établir la validité de notre algorithme :

Proposition 20. *L'algorithme Ω-Filter retourne exactement l'ensemble* $\mathrm{PO}_\Omega(X)$.

Démonstration. Notons X_m le sous-ensemble de X retourné par l'algorithme Ω-Filter. Montrons que nous avons $\mathrm{PO}_\Omega(X) \subseteq X_m$ et $X_m \subseteq \mathrm{PO}_\Omega(X)$.

• $\mathrm{PO}_\Omega(X) \subseteq X_m$: soit $x^i \in \mathrm{PO}_\Omega(X)$. Par définition de $\mathrm{PO}_\Omega(X)$, il existe $\omega' \in \Omega$ tel que $x^i \in \arg\min_{x \in X} f_{\omega'}(x)$; autrement dit, nous avons $f_{\omega'}(x) \geq f_{\omega'}(x^i)$ pour tout $x \in X$. Par conséquent, nous avons $\neg(X \prec_\Omega x^i)$ par définition 47. De ce fait, nous avons $\neg(X_{1-1}^\sigma \prec_\Omega x^{\sigma(1)})$ pour toute permutation σ de $\{1, \ldots, m\}$ telle que $\sigma(1) = i$; ainsi, nous avons forcément $x^i \in X_m^\sigma$ par construction. En utilisant la proposition 19, nous en déduisons finalement que x^i appartient à $X_m^\sigma = X_m$.

• $X_m \subseteq \mathrm{PO}_\Omega(X)$: soit $x^i \in X_m$ et σ une permutation de $\{1, \ldots, m\}$ telle que $\sigma(1) = i$. Puisque $x^i \in X_m$, alors nous avons $x^i \in X_m = X_m^\sigma$ d'après la proposition 19. Par construction de X_m^σ, nous en déduisons $\neg(X_{1-1}^\sigma \prec_\Omega x^i)$; autrement dit, nous avons $\neg(X \prec_\Omega x^i)$. Par conséquent, il existe $\omega' \in \Omega$ tel que $f_{\omega'}(x) \geq f_{\omega'}(x^i)$ pour tout $x \in X$. D'où $x^i \in \mathrm{PO}_\Omega(X)$. □

Finalement, nous avons proposé un algorithme permettant de calculer $\mathrm{PO}_\Omega(X)$ en temps polynomial pour un ensemble fini de vecteurs X donné. Néanmoins, dans le cadre de la problématique de recherche dans un graphe d'états, appliquer cet algorithme sur l'ensemble $X = \mathrm{ND}(\mathcal{X})$ ne serait pas très efficace en pratique car ce dernier ensemble peut contenir un nombre exponentiel de vecteurs (par rapport au nombre de nœuds du graphe). Par conséquent, nous proposons à la place une variante de l'algorithme de MOA* retournant directement l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$.

ii) Un algorithme par programmation dynamique

Puisque la plupart des modèles de préférences considérés en décision multicritère sont monotones par rapport à la dominance de Pareto, alors MOA* représente souvent une base utile pour le développement de méthodes d'exploration plus spécifiques. En particulier, il s'agit pour nous de déterminer l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ des vecteurs coût potentiellement optimaux et un chemin par élément de cet ensemble. Comme les vecteurs coût potentiellement optimaux au sens d'une somme pondérée sont nécessairement Pareto-optimaux, alors nous proposons d'étendre l'algorithme MOA* en ajoutant de nouvelles règles d'élagage. Celles-ci vont nous permettre de supprimer en cours de résolution des chemins menant à des solutions

Pareto-optimales qui ne sont pas potentiellement optimales. Nous présentons maintenant les grandes lignes de notre algorithme.

Comme l'algorithme MOA* [Mandow and Pérez de la Cruz, 2005b], notre procédure explore le graphe G en développant des étiquettes correspondant à des chemins du graphe. Ces étiquettes sont de la forme $\ell = [n_\ell, p_\ell, g_\ell]$ où p_ℓ est un chemin du nœud source s au nœud n_ℓ et $g_\ell = g(p_\ell)$ est son vecteur coût. À chaque itération, une étiquette ℓ^* est choisie pour être développée, ce qui consiste à engendrer l'ensemble de ses chemins successeurs $\{[n, p_{\ell^*} \circ n, g(p_{\ell^*} \circ n)] : n \in \Pi(n_{\ell^*})\}$, où $\circ$ représente l'opérateur de concaténation de chemins. Nous distinguons alors deux types d'étiquettes engendrées : l'ensemble C des étiquettes déjà développées (*Closed*) et l'ensemble O des étiquettes candidates pour le prochain développement (*Open*). L'ensemble C (resp. O) restreint aux étiquettes ℓ telles que $p_\ell \in P(s, n)$ est noté $C(n)$ (resp. $O(n)$) ; en particulier, nous noterons S l'ensemble $\bigcup_{\gamma\in\Gamma} C(\gamma)$ composé des étiquettes déjà développées et correspondant à des chemins solutions. Pour chaque étiquette engendrée ℓ, un ensemble de vecteurs coût $F(\ell) = \{g_\ell + h : h \in H(n_\ell)\}$ est calculé, où $H(n_\ell)$ est un ensemble de vecteurs heuristiques estimant l'ensemble $\{g(p) : p \in P(n_\ell, \Gamma)\}$. Comme dans l'algorithme MOA*, les ensembles $F(\ell)$ vont nous servir à choisir les prochaines étiquettes à développer et à supprimer des étiquettes non pertinentes en cours de résolution. Notre algorithme se distingue de l'algorithme MOA* en utilisant des règles de suppression d'étiquettes plus discriminantes que la dominance de Pareto. Plus précisément, notre algorithme repose sur la propriété suivante :

Proposition 21 (Additivité)**.** *Pour tout $y \in \mathbb{R}^q_+$ et tout $Z \subseteq \mathbb{R}^q_+$: $Z \prec_\Omega y \Rightarrow \forall x \in \mathbb{R}^q_+$, $Z \oplus \{x\} \prec_\Omega y + x$, où $B \oplus C = \{b + c : b \in B, c \in C\}$ pour tous $B, C \subseteq \mathbb{R}^q_+$.*

Démonstration. Soient $y \in \mathbb{R}^q_+$ et $Z \subseteq \mathbb{R}^q_+$ tels que $Z \prec_\Omega y$. Soient $x \in \mathbb{R}^q_+$ et $\omega \in \Omega$. Comme $Z \prec_\Omega y$ par hypothèse, alors il existe $z \in Z$ tel que $f_\omega(z) < f_\omega(y)$. Nous obtenons alors $f_\omega(z) + f_\omega(x) < f_\omega(y) + f_\omega(x)$ par additivité. Puisque la fonction f_ω est une somme pondérée, alors nous en déduisons $f_\omega(z + x) < f_\omega(y + x)$. Finalement, comme $z + x \in Z \oplus \{x\}$, alors nous avons bien $Z \oplus \{x\} \prec_\Omega y + x$. □

Cette proposition nous informe que, durant la recherche, nous pouvons éliminer une étiquette $\ell' \in O(n)$ s'il existe un sous-ensemble $L \subseteq O(n) \cup C(n)$ tel que $\{g_\ell : \ell \in L\} \prec_\Omega g_{\ell'}$. Dans ce cas, toute extension du chemin $p_{\ell'}$ sera en effet Ω-dominé par la même extension des chemins $\{p_\ell : \ell \in L\}$, ce qui implique que tout chemin solution issu de $p_{\ell'}$ n'est pas potentiellement optimal (cf. proposition 17). Ainsi, pour limiter la taille de l'arbre d'exploration, il convient de supprimer toutes les étiquettes $\ell' \in O(n)$ tels que $\{g_\ell : \ell \in L\} \prec_\Omega g_{\ell'}$ pour un sous-ensemble $L \subseteq O(n) \cup C(n)$. D'après la proposition 17, ceci revient à éliminer toutes les étiquettes $\ell' \in O(n)$ telles que $g_{\ell'} \notin \mathrm{PO}_\Omega(\{g_\ell : \ell \in O(n) \cup C(n)\})$. Par conséquent, nous proposons de supprimer des étiquettes par application de la règle d'élagage suivante :

Règle R1. *Supprimer les étiquettes $\ell' \in O(n)$ telles que $g_{\ell'} \notin \Omega$-Filter$(\{g_\ell : \ell \in O(n) \cup C(n)\})$.*

La seconde règle d'élagage que nous utilisons suppose, comme dans MOA*, que l'heuristique H est admissible ; autrement dit, H nous donne une évaluation optimiste des chemins arrivant à un nœud but. Cette hypothèse se formalise comme suit : pour tout nœud $n \in N$ et pour tout chemin $p \in P(n, \Gamma)$, il existe un vecteur $h \in H(n)$ tel que $h \precsim_P g(p)$. Sous cette hypothèse, nous avons le résultat suivant :

Proposition 22. *Pour toute étiquette $\ell' \in O$, si $\{g_\ell : \ell \in S \cup \bigcup_{\gamma\in\Gamma} O(\gamma)\} \prec_\Omega f'$ pour tout $f' \in F(\ell')$, alors tout chemin solution issu de $p_{\ell'}$ n'est pas potentiellement optimal.*

Démonstration. Posons $L = S \cup \bigcup_{\gamma\in\Gamma} O(\gamma)$. Soit $\ell' \in O$ telle que $\{g_\ell : \ell \in L\} \prec_\Omega f'$ pour tout $f' \in F(\ell')$. Pour tout chemin $p \in P(n_{\ell'}, \Gamma)$, nous voulons prouver que $g(p_{\ell'} \circ p) \notin \mathrm{PO}_\Omega(\mathcal{X})$, où $\circ$ représente l'opérateur de concaténation de chemins. Puisque H est admissible, alors il existe $h \in H(n_{\ell'})$ tel que $h \precsim_P g(p)$, ce qui nous permet de déduire $h + g_{\ell'} \precsim_P g(p) + g_{\ell'}$ par additivité de la dominance de Pareto ; ainsi, il existe $f' \in F(\ell')$ tel que $f' \precsim_P g(p) + g_{\ell'} = g(p_{\ell'} \circ p)$. Pour tout $\omega \in \Omega$, nous avons alors $f_\omega(f') \leq f_\omega(g(p_{\ell'} \circ p))$ puisque f_ω est une fonction croissante en ses composantes. Par ailleurs, nous avons $\{g_\ell : \ell \in L\} \prec_\Omega f'$, ce qui implique $f_\omega(x) < f_\omega(f')$ pour au moins un vecteur $x \in \{g_\ell : \ell \in L\}$. Nous obtenons ensuite $f_\omega(x) < f_\omega(g(p_{\ell'} \circ p))$ par transitivité, ce qui implique $\{g_\ell : \ell \in L\} \prec_\Omega g(p_{\ell'} \circ p)$. Enfin, comme $g(p_{\ell'} \circ p) \in \mathcal{X}$ et $\{g_\ell : \ell \in L\} \subseteq \mathcal{X}$, alors nous en déduisons $g(p_{\ell'} \circ p) \notin \mathrm{PO}_\Omega(\mathcal{X})$ (cf. proposition 17). □

Ainsi, nous proposons la règle suivante :

Règle R2. *Supprimer l'étiquette $\ell' \in O$ si $\{g_\ell : \ell \in S \cup \bigcup_{\gamma\in\Gamma} O(\gamma)\} \prec_\Omega f'$ pour tout $f' \in F(\ell')$.*

Finalement, notre algorithme se résume par l'algorithme 4, où Choisir(O) est une procédure consistant à retourner une étiquette $\ell \in O$ vérifiant la propriété suivante : $\exists f \in F(\ell), \forall \ell' \in O, \forall f' \in F(\ell'), \neg(f' \prec_P f)$.

Algorithm 4:

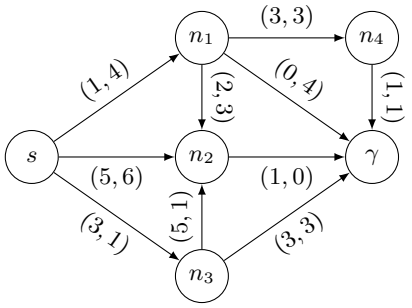
Input: $G = (N, A)$; s ; Γ ; H ; Ω
$O \leftarrow \{[s, \langle s\rangle, (0, \ldots, 0)]\}$
$S,\ C \leftarrow \emptyset$
while $O \neq \emptyset$ **do**
  $\ell^* \leftarrow$ Choisir(O)
  Déplacer l'étiquette ℓ^* de l'ensemble des ouverts O vers l'ensemble des fermés C
  if $n_{\ell^*} \in \Gamma$ **then**
    Ajouter l'étiquette ℓ^* à l'ensemble de solutions S
  else
    foreach $n \in \Pi(n_{\ell^*})$ **do**
      Générer l'étiquette $\ell' = [n, p_{\ell^*} \circ \langle n\rangle, g_{\ell^*} + g(n_{\ell^*}, n)]$
      if *pour tout* $\ell \in O(n) \cup C(n)$, $\neg(g_\ell \precsim_P g_{\ell'})$ **then**
        Ajouter l'étiquette ℓ' à l'ensemble des ouverts O
        Appliquer la règle d'élagage R2 à l'étiquette ℓ'
        if ℓ' *n'a pas été supprimée* **then**
          Appliquer la règle d'élagage R1 à l'ensemble des ouverts $O(n)$
        end
      end
    end
  end
end
return une étiquette $\ell \in S$ par élément de l'ensemble Ω-Filter($\{g_{\ell'} : \ell' \in S\}$)

Proposition 23. *L'algorithme 4 détermine exactement l'ensemble* $\mathrm{PO}_\Omega(\mathcal{X})$ *et un chemin par élément de cet ensemble.*

Démonstration. Tout d'abord, rappelons que cet algorithme est fondé sur MOA* qui est un algorithme permettant de déterminer un chemin solution par vecteur coût Pareto-optimal. Ce dernier est ici raffiné par les règles d'élagage R1 et R2. Grâce aux propositions 21 et 22, nous savons que ces deux règles ne peuvent pas éliminer des chemins conduisant à des chemins solutions potentiellement optimaux. Par conséquent, à la fin de la boucle "while", l'ensemble S contient forcément tous les éléments de $\mathrm{PO}_\Omega(\mathcal{X})$. Ainsi, le dernier appel à Ω-Filter sur l'ensemble S permet d'éliminer, si nécessaire, tous les vecteurs coûts qui ne sont pas dans $\mathrm{PO}_\Omega(\mathcal{X})$. □

Pour illustrer notre algorithme, nous décrivons ci-dessous son exécution sur un graphe bicritère :

Exemple 23. *Considérons le graphe* $G = (N, A)$*, avec* $N = \{s, n_1, n_2, n_3, \gamma\}$*, représenté ci-dessous :*



Sur cette figure, le vecteur coût $g(a)$ *est indiqué au milieu de chaque arc* $a \in A$*. Pour simplifier l'exemple, nous considérons uniquement l'évaluation heuristique nulle ; autrement dit, nous avons* $H(n) = \{(0, 0)\}$ *pour tout* $n \in N$*. Supposons que l'ensemble des données* $\mathcal{P}$ *est vide. Dans ce cas, nous avons* $\Omega = \{(\omega_1, \omega_2) \in int(\mathbb{R}^2_+) : \omega_1 + \omega_2 = 1\}$ *où int représente l'intérieur du cône. Initialement, nous avons* $O = \{\ell_s\}$ *où* $\ell_s = [s, \langle s \rangle, (0, 0)]$*. Durant la première itération, l'étiquette* ℓ_s *est déplacée de l'ensemble* O *vers l'ensemble* C *pour être développée. Comme nous avons* $\Pi(s) = \{n_1, n_2, n_3\}$*, alors les trois étiquettes suivantes sont engendrées :* $\ell_1 = [n_1, \langle s, n_1 \rangle, (1, 4)]$*,* $\ell_2 = [n_2, \langle s, n_2 \rangle, (5, 6)]$ *et* $\ell_3 = [n_3, \langle s, n_3 \rangle, (3, 1)]$*. Ces étiquettes sont toutes insérées dans l'ensemble* O *puisque* $O(n_i) \cup C(n_i) = \emptyset$ *pour tout* $i \in \{1, 2, 3\}$ *et* $S \cup O(\gamma) = \emptyset$*. Ces dernières étiquettes sont donc les trois seules candidates au prochain développement. Notons que nous avons* $F(\ell_1) = \{(1, 4)\}$*,* $F(\ell_2) = \{(5, 6)\}$ *et* $F(\ell_3) = \{(3, 1)\}$*. Comme* $(5, 6)$ *est le seul vecteur Pareto-dominé, Choisir*(O) *peut retourner* ℓ_1 *ou* ℓ_3*. Supposons que Choisir*(O) *retourne* ℓ_3*. Dans ce cas, durant la deuxième itération, l'algorithme déplace* ℓ_3 *de l'ensemble* O *vers l'ensemble* C *pour développer cette étiquette. Comme* $\Pi(n_3) = \{n_2, \gamma\}$*, alors ce développement engendre les étiquettes* $\ell'_2 = [n_2, \langle s, n_3, n_2 \rangle, (8, 2)]$ *et* $\ell_\gamma = [\gamma, \langle s, n_3, \gamma \rangle, (6, 4)]$*. L'algorithme réalise alors le traitement suivant :*

- *Étiquette* ℓ'_2 *: rappelons que* $O(n_2) = \{\ell_2\}$ *et* $C(n_2) = \emptyset$*. De ce fait, comme* $\neg(g_{\ell_2} \precsim_P g_{\ell'_2})$*, alors* ℓ'_2 *est insérée dans l'ensemble* O *(cf. ligne 12). Par ailleurs, comme* $S \cup O(\gamma) = \emptyset$*, alors la règle R2 ne supprime pas* ℓ'_2 *(cf. ligne 13). Enfin, puisque* $\mathrm{PO}_\Omega(\{g_{\ell_2}, g_{\ell'_2}\}) = \{g_{\ell_2}, g_{\ell'_2}\}$*, alors aucune étiquette n'est retirée de l'ensemble* $O(n_2)$ *par application de la règle R1 (cf. ligne 15).*

- *Étiquette ℓ_γ : comme $O(\gamma) \cup S = \emptyset$, alors l'étiquette ℓ_γ est insérée dans O et ℓ_γ n'est pas éliminée par les règles R1 et R2.*

Finalement, nous avons $O = \{\ell_1, \ell_2, \ell'_2, \ell_\gamma\}$ à la fin de la deuxième itération. Nous avons $F(\ell_1) = \{(1,4)\}$, $F(\ell_2) = \{(5,6)\}$, $F(\ell'_2) = \{(8,2)\}$ et $F(\ell_\gamma) = \{(6,4)\}$. Comme $(5,6)$ est le seul vecteur Pareto-dominé, alors Choisir(O) peut retourner ℓ_1, ℓ'_2 ou ℓ_γ pour la troisième itération. Supposons que Choisir(O) retourne ℓ_γ. Dans ce cas, ℓ_γ est déplacée de O vers C puis ℓ_γ est insérée dans l'ensemble S (cf. ligne 7). Durant la quatrième itération, Choisir(O) peut retourner ℓ_1 ou ℓ'_2. Supposons que Choisir(O) retourne ℓ_1. Dans ce cas, ℓ_1 est déplacée de O vers C. Comme $\Pi(n_1) = \{n_2, \gamma, n_4\}$, alors les étiquettes suivantes sont créées : $\ell''_2 = [n_2, \langle s, n_1, n_2\rangle, (3,7)]$, $\ell'_\gamma = [\gamma, \langle s, n_1, \gamma\rangle, (1,8)]$ et $\ell_4 = [n_4, \langle s, n_1, n_4\rangle, (4,7)]$. Ces étiquettes sont ensuite traitées comme décrit ci-dessous :

- *Étiquette ℓ''_2 : rappelons que $O(n_2) = \{\ell_2, \ell'_2\}$ et $C(n_2) = \emptyset$. De ce fait, comme $\neg(g_{\ell_2} \precsim_P g_{\ell''_2})$ et $\neg(g_{\ell'_2} \precsim_P g_{\ell''_2})$, alors ℓ''_2 est insérée dans l'ensemble O (cf. ligne 12). Par ailleurs, ℓ''_2 n'est pas éliminée par la règle R2 car nous avons $\neg(\{g_{\ell_\gamma}\} \prec_\Omega g_{\ell''_2})$ (cf. ligne 13). Néanmoins, ℓ_2 est ensuite supprimée par la règle R1 puisque $\mathrm{PO}_\Omega(\{g_{\ell_2}, g_{\ell'_2}, g_{\ell''_2}\}) = \{g_{\ell'_2}, g_{\ell''_2}\}$ (cf. ligne 15).*
- *Étiquette ℓ'_γ : rappelons que $O(\gamma) = \emptyset$ et $S = \{\ell_\gamma\}$. Par conséquent, comme $\neg(g_{\ell_\gamma} \precsim_P g_{\ell'_\gamma})$, alors ℓ'_γ est insérée dans O (cf. ligne 12). Puis, la règle R2 ne supprime pas ℓ'_γ car $\neg(\{g_{\ell_\gamma}, g_{\ell'_\gamma}\} \prec_\Omega g_{\ell'_\gamma})$ (cf. ligne 13). Enfin, la règle R1 ne supprime rien car $\mathrm{PO}_\Omega(\{g_{\ell_\gamma}, g_{\ell'_\gamma}\}) = \{g_{\ell_\gamma}, g_{\ell'_\gamma}\}$ (cf. ligne 15).*
- *Étiquette ℓ_4 : comme $O(n_4) \cup C(n_4) = \emptyset$, alors l'étiquette ℓ_4 est insérée dans O. Toutefois, cette étiquette est ensuite supprimée par la règle R2 car nous avons $\{g_{\ell_\gamma}, g_{\ell'_\gamma}\} \prec_\Omega g_{\ell_4}$.*

Finalement, nous avons $O = \{\ell'_2, \ell''_2, \ell'_\gamma\}$ à la fin de la quatrième itération. Notons que nous avons $F(\ell'_2) = \{(8,2)\}$, $F(\ell''_2) = \{(3,7)\}$ et $F(\ell'_\gamma) = \{(1,8)\}$. Comme aucun de ces vecteurs n'est Pareto-dominé, alors Choisir(O) peut retourner ℓ'_2, ℓ''_2 ou ℓ'_γ à la cinquième itération. Supposons que Choisir(O) retourne ℓ'_γ : cette étiquette est alors retirée de O et insérée dans S (cf. ligne 7). Pour la sixième itération, Choisir(O) peut retourner ℓ'_2 ou ℓ''_2. Supposons que Choisir(O) retourne ℓ''_2. Dans ce cas, ℓ''_2 est déplacée de O vers C. Comme $\Pi(n_2) = \{\gamma\}$, alors l'étiquette suivante est engendrée : $\ell''_\gamma = [\gamma, \langle s, n_1, n_2, \gamma\rangle, (4,7)]$. Rappelons que $O(\gamma) = \emptyset$ et $S = \{\ell_\gamma, \ell'_\gamma\}$. De ce fait, comme $\neg(g_{\ell_\gamma} \precsim_P g_{\ell''_\gamma})$ et $\neg(g_{\ell'_\gamma} \precsim_P g_{\ell''_\gamma})$, alors ℓ''_γ est insérée dans O (cf. ligne 12). Cependant, la règle R2 élimine ensuite cette étiquette car nous avons $\{g_{\ell_\gamma}, g_{\ell'_\gamma}, g_{\ell''_\gamma}\} \prec_\Omega g_{\ell''_\gamma}$ (cf. ligne 13). Ainsi, nous avons $O = \{\ell'_2\}$ à la fin de cette itération. Par conséquent, l'algorithme développe ℓ'_2 à la septième itération. Cette étiquette est donc tout d'abord déplacée de O vers C. Puis, comme $\Pi(n_2) = \{\gamma\}$, alors l'étiquette suivante est créée : $\ell'''_\gamma = [\gamma, \langle s, n_3, n_2, \gamma\rangle, (9,2)]$. À cette itération, nous avons $O(\gamma) = \emptyset$ et $S = \{\ell_\gamma, \ell'_\gamma\}$. De ce fait, comme $\neg(g_{\ell_\gamma} \precsim_P g_{\ell'''_\gamma})$ et $\neg(g_{\ell'_\gamma} \precsim_P g_{\ell'''_\gamma})$, alors ℓ'''_γ est insérée dans O (cf. ligne 12). Puis, les règles R1 et R2 ne suppriment pas cette étiquette car nous avons $\mathrm{PO}_\Omega(\{g_{\ell_\gamma}, g_{\ell'_\gamma}, g_{\ell'''_\gamma}\}) = \{g_{\ell_\gamma}, g_{\ell'_\gamma}, g_{\ell'''_\gamma}\}$. Ainsi, nous avons $O = \{\ell'''_\gamma\}$ à la fin de la septième itération. La huitième itération correspond donc au développement de ℓ'''_γ : cette étiquette est alors retirée de l'ensemble O et insérée dans l'ensemble S (cf. ligne 7). Finalement, nous avons $O = \emptyset$ à la fin de cette itération. Comme $\mathrm{PO}_\Omega(\{g_\ell : \ell \in S\}) = \{g_\ell : \ell \in S\}$, alors l'algorithme retourne les trois étiquettes contenues dans l'ensemble S à la fin de son exécution (cf. ligne 21). Ces trois étiquettes représentent donc les seuls vecteurs solutions potentiellement optimaux dans ce problème.

Dans cette sous-section, nous avons proposé une extension de l'algorithme MOA* permettant de déterminer l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ de manière efficace en utilisant la programmation dynamique. Cependant, il existe des instances de graphes pour lesquelles tous les chemins solutions sont potentiellement optimaux et sont en nombre exponentiel (e.g., [Hansen, 1980]) ; dans ces situations, l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ ne peut évidemment pas être énuméré en temps polynomial. Cette observation nous suggère de poser des questions au décideur pour réduire Ω dans l'espoir de diminuer significativement le nombre d'éléments de $\mathrm{PO}_\Omega(\mathcal{X})$; en effet, nous avons $\mathrm{PO}_{\Omega'}(\mathcal{X}) \subseteq \mathrm{PO}_\Omega(\mathcal{X})$ pour tout $\Omega' \subseteq \Omega$. Ceci est l'objet de la sous-section suivante.

3.1.3 Des algorithmes interactifs par programmation dynamique pour déterminer une solution optimale avec la somme pondérée

Dans la sous-section précédente, nous avons considéré que l'ensemble Ω des jeux de poids admissibles était stable au cours du temps. À présent, nous considérons que de nouvelles informations sur les préférences du décideur sont obtenues à différents pas de temps notés $1, \ldots, T$, en supposant que notre algorithme commence au temps 0 et se termine au temps $T+1$. Ces données induisent de nouvelles contraintes sur les jeux de poids admissibles, réduisant ainsi progressivement la taille de l'ensemble Ω en cours de résolution ; notons Ω_t l'ensemble Ω au pas de temps t de la résolution. Par définition, ces derniers forment une séquence d'ensembles emboîtés : $\Omega_{t+1} \subseteq \Omega_t$ pour tout $t \in \{0, \ldots, T\}$. De ce fait, nous avons $\mathrm{PO}_{\Omega_{t+1}}(\mathcal{X}) \subseteq \mathrm{PO}_{\Omega_t}(\mathcal{X})$ pour tout $t \in \{0, \ldots, T\}$ (cf. définition 46). Par conséquent, toute étiquette supprimée en considérant l'ensemble Ω_t devrait aussi être supprimée si on considérait $\Omega_{t'}$ à la place, avec $t' \in \{t+1, \ldots, T+1\}$. Ainsi, si l'ensemble Ω réduit en cours de résolution, alors il n'est pas nécessaire de relancer la procédure car les suppressions antérieures ne sont pas remises en cause, la dernière instruction assurant ensuite que l'algorithme retourne exactement l'ensemble $\mathrm{PO}_{\Omega_{T+1}}(\mathcal{X})$ (cf. ligne 21). Cette observation nous informe qu'il est possible d'interagir avec le décideur durant la recherche pour centrer plus rapidement la recherche sur les chemins préférés.

Idéalement, il faudrait poser des questions de manière à garantir que l'ensemble $\mathrm{PO}_{\Omega_{T+1}}(\mathcal{X})$ contienne un vecteur solution *nécessairement optimal*, c'est-à-dire un vecteur minimisant f_ω pour tout $\omega \in \Omega_{T+1}$. En même temps, il convient de limiter le nombre de questions posées au décideur pour minimiser l'effort d'élicitation. Ces deux objectifs concurrentiels nous suggèrent d'utiliser l'approche incrémentale fondée sur le critère Minimax Regret. En Section 1.4.2, cette approche a été présentée et discutée dans le cadre d'une fonction f_ω à maximiser. Pour obtenir son équivalent en minimisation, il convient d'utiliser les définitions suivantes :

Définition 48. *Pour tous $x, y \in \mathcal{X}$:*

$$\begin{aligned} \mathrm{PMR}(x, y, \Omega) &= \max_{\omega \in \Omega} \{f_\omega(x) - f_\omega(y)\} \\ \mathrm{MR}(x, \mathcal{X}, \Omega) &= \max_{y \in \mathcal{X}} \mathrm{PMR}(x, y, \Omega) \\ \mathrm{mMR}(\mathcal{X}, \Omega) &= \min_{x \in \mathcal{X}} \mathrm{MR}(x, \mathcal{X}, \Omega) \end{aligned}$$

Rappelons que cette approche d'élicitation est fondée sur le calcul répété de la valeur $\mathrm{PMR}(x, y, \Omega)$ pour toute paire (x, y) de solutions possibles, tant que le regret $\mathrm{mMR}(\mathcal{X}, \Omega)$ est strictement plus grand

que la valeur $\delta \geq 0$ représentant un niveau d'erreur que le décideur juge acceptable (avec $\delta = 0$, on obtient à la fin une solution nécessairement optimale). Dans notre contexte, cette approche ne peut pas être mise en œuvre de cette façon car l'ensemble $\mathcal{X}$ des vecteurs solutions est de trop grande taille. À la place, nous allons intégrer l'élicitation à la recherche réalisée par l'algorithme 4 pour travailler sur des ensembles de solutions plus réduits tout en garantissant de détecter une solution nécessairement optimale à la fin de l'exécution. Plus précisément, nous proposons les stratégies suivantes :

Stratégie S1. Il s'agit de poser des questions uniquement lorsque la valeur $\text{mMR}(\{g_\ell : \ell \in S\}, \Omega)$ devient strictement positive suite à l'insertion d'une nouvelle étiquette dans S. Dans ces situations, S1 interroge le décideur tant que $\text{mMR}(\{g_\ell : \ell \in S\}, \Omega) > 0$. Pour engendrer des questions, nous pouvons par exemple suivre la stratégie de sélection CSS présentée en Section 1.4.2. Montrons que S1 permet de détecter un vecteur solution nécessairement optimal à la fin de l'algorithme. Rappelons que l'algorithme 4 combiné avec la stratégie S1 retourne une étiquette $\ell \in S$ par vecteur coût de l'ensemble $\text{PO}_{\Omega_{T+1}}(\{g_\ell : \ell \in S\})$ (cf. ligne 21), celui-ci étant égal à l'ensemble $\text{PO}_{\Omega_{T+1}}(\mathcal{X})$. Par définition, les solutions préférées sont forcément dans cet ensemble. Il s'agit donc de montrer que nous avons une étiquette $\ell' \in S$ telle que $f_\omega(g_{\ell'}) \leq f_\omega(g_\ell)$ pour tout jeu de poids $\omega \in \Omega_{T+1}$ et toute étiquette $\ell \in S$. Comme la stratégie $S1$ assure que $\text{mMR}(\{g_\ell : \ell \in S\}, \Omega_{T+1}) \leq 0$, alors il existe $\ell' \in S$ telle que $\text{MR}(g_{\ell'}, \{g_\ell : \ell \in S\}, \Omega_{T+1}) \leq 0$. Par définition de la valeur MR, nous avons donc $\text{PMR}(\ell', \ell, \Omega) \leq 0$ pour tout $\ell' \in S$, ce qui signifie que l'inégalité $f_\omega(g_{\ell'}) \leq f_\omega(g_\ell)$ est vraie pour tout $\omega \in \Omega_{T+1}$; ceci permet de conclure la preuve.

Stratégie S2. Cette stratégie a pour objectif de mieux diriger l'exploration du graphe en sélectionnant plus pertinemment la prochaine étiquette à développer ; autrement dit, la procédure Choisir est ici modifiée pour se concentrer sur les solutions les plus prometteuses. Cette étape de sélection est maintenant réalisée en posant des questions au décideur tant que la valeur $\text{mMR}(X, \Omega)$ est strictement positive, où $X = \{g_\ell + h : \ell \in O, h \in H(n_\ell)\}$. La sélection des questions peut par exemple être dirigée par la stratégie CSS (cf. Section 1.4.2). La prochaine étiquette à développer est ensuite choisie parmi les étiquettes $\ell \in O$ telles que le $\text{MR}(g_\ell + h, X, \Omega) = 0$ pour au moins un vecteur heuristique $h \in H(n_\ell)$. Utiliser cette stratégie revient donc à développer uniquement des sous-chemins ayant au moins un vecteur heuristique nécessairement optimal dans X. Par ailleurs, si $\text{mMR}(\{g_\ell : \ell \in S\}, \Omega) > 0$ à la fin de l'exécution de l'algorithme, alors $S2$ pose des questions tant que cette inégalité reste vraie. De manière similaire à la stratégie S1, on peut montrer que cette étape finale permet de garantir que l'algorithme retourne une solution nécessairement optimale à la fin de son exécution.

Dans le cadre de ces deux stratégies, le décideur compare uniquement des vecteurs coût associés à des chemins solutions, ce qui a du sens. Ces vecteurs représentent des coûts réels dans S1 et des évaluations heuristiques dans S2. Ainsi, pour la stratégies S1, les questions peuvent être accompagnées de la présentation des chemins à comparer pour aider le décideur dans sa prise de décision. En guise d'illustration, nous présentons ci-dessous une exécution de l'algorithme 4 combiné avec la stratégie S1 :

Exemple 24. *Reprenons l'exemple 23 en supposant que les préférences du décideur sont représentables par la somme pondérée de jeu de poids* $\omega_0 = (0.6, 0.4)$. *Sans aucune information sur ses préférences,*

l'ensemble Ω *des jeux de poids* $\omega = (\omega_1, \omega_2)$ *admissibles est décrit par la contrainte* $0 < \omega_1 < 1$*, le poids* ω_2 *étant défini de manière implicite par la contrainte de normalisation des poids (i.e.* $\omega_2 = 1 - \omega_1$*).*

Déroulons l'algorithme 4 combiné avec S1 (utilisant la stratégie CSS). Comme S1 engendre des questions uniquement quand la valeur $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega)$ *devient strictement positive suite à l'insertion d'une étiquette dans* S*, alors les quatre premières itérations sont ici identiques à celles de la version non interactive. Reprenons l'exécution à partir de là. Au début de la cinquième itération, nous avons :*

- $O = \{\ell'_2, \ell''_2, \ell'_\gamma\}$ *où* $\ell'_2 = [n_2, \langle s, n_3, n_2\rangle, (8, 2)]$*,* $\ell''_2 = [n_2, \langle s, n_1, n_2\rangle, (3, 7)]$*,* $\ell'_\gamma = [\gamma, \langle s, n_1, \gamma\rangle, (1, 8)]$*.*
- $C = \{\ell_s, \ell_1, \ell_3, \ell_\gamma\}$*, où* $\ell_s = [s, \langle s\rangle, (0, 0)]$*,* $\ell_1 = [n_1, \langle s, n_1\rangle, (1, 4)]$*,* $\ell_3 = [n_3, \langle s, n_3\rangle, (3, 1)]$*,* $\ell_\gamma = [\gamma, \langle s, n_3, \gamma\rangle, (6, 4)]$*. Ainsi, nous avons* $S = \{\ell_\gamma\}$*.*

De plus, nous avons $F(\ell'_2) = \{(8, 2)\}$*,* $F(\ell''_2) = \{(3, 7)\}$ *et* $F(\ell'_\gamma) = \{(1, 8)\}$*. Comme aucun de ces vecteurs n'est Pareto-dominé, alors Choisir*(O) *peut retourner* ℓ'_2*,* ℓ''_2 *ou* ℓ'_γ*. Supposons que Choisir*(O) *retourne l'étiquette* ℓ'_γ*. Dans ce cas,* ℓ'_γ *est retirée de l'ensemble* O *et insérée dans l'ensemble* S *(cf. ligne 7). Suite à cette insertion, nous obtenons* $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega) > 0$*. La stratégie S1 demande alors au décideur de comparer les vecteurs coût* $(6, 4)$ *et* $(1, 8)$ *associés respectivement aux chemins* $\langle s, n_3, \gamma\rangle$ *et* $\langle s, n_1, \gamma\rangle$ *(question engendrée par la stratégie CSS). Comme* $f_{\omega_0}(6, 4) = 5.2 \geq f_{\omega_0}(1, 8) = 3.8$*, alors le décideur nous apprend qu'il préfère le second chemin au premier. Cette information induit la contrainte linéaire* $f_\omega(6, 4) = 6\omega_1 + 4(1 - \omega_1) \geq f_\omega(1, 8) = \omega_1 + 8(1 - \omega_1)$ *sur l'espace des paramètres admissibles, qui est équivalente à la contrainte* $\omega_1 \geq 4/9$*. Ainsi,* $\Omega = \{(\omega_1, 1 - \omega_1) \in \mathbb{R}^2_+ : 4/9 \leq \omega_1 < 1\}$*. Après cette mise à jour, nous obtenons* $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega) = \mathrm{MR}((1, 8), \{g_\ell : \ell \in S\}, \Omega) \leq 0$*. Par conséquent, S1 ne produit plus aucune question à cette itération ; le vecteur solution* $(1, 8)$ *constitue la meilleure option trouvée jusque là. Pour la sixième itération, Choisir*(O) *peut retourner* ℓ'_2 *ou* ℓ''_2*. Supposons que Choisir*(O) *retourne* ℓ''_2*. Dans ce cas,* ℓ''_2 *est déplacée de* O *vers* C*. Comme* $\Pi(n_2) = \{\gamma\}$*, alors l'étiquette suivante est engendrée :* $\ell''_\gamma = [\gamma, \langle s, n_1, n_2, \gamma\rangle, (4, 7)]$*. L'étiquette* ℓ''_2 *est ensuite insérée dans* O *puisque* $\neg(g_{\ell_\gamma} \precsim_P g_{\ell''_\gamma})$ *et* $\neg(g_{\ell'_\gamma} \precsim_P g_{\ell''_\gamma})$ *(cf. ligne 12). Néanmoins, cette étiquette est ensuite supprimée par la règle R2 car nous avons* $\{g_{\ell_\gamma}, g_{\ell'_\gamma}, g_{\ell''_\gamma}\} \prec_\Omega g_{\ell''_\gamma}$ *(cf. ligne 13). Ainsi, nous avons* $O = \{\ell'_2\}$ *à la fin de cette itération. Par conséquent, l'algorithme développe forcément l'étiquette* ℓ'_2 *durant la septième itération, en commençant par déplacer celle-ci de* O *vers* C*. Puis, comme* $\Pi(n_2) = \{\gamma\}$*, alors l'étiquette suivante est créée :* $\ell'''_\gamma = [\gamma, \langle s, n_3, n_2, \gamma\rangle, (9, 2)]$*. L'étiquette* ℓ'''_γ *est ensuite insérée dans* O *car nous avons* $\neg(g_{\ell_\gamma} \precsim_P g_{\ell'''_\gamma})$ *et* $\neg(g_{\ell'_\gamma} \precsim_P g_{\ell'''_\gamma})$ *(cf. ligne 12). Cependant, cette étiquette est ensuite éliminée par la règle R2 car* $\{g_{\ell_\gamma}, g_{\ell'_\gamma}, g_{\ell'''_\gamma}\} \prec_\Omega g_{\ell'''_\gamma}$ *(cf. ligne 13). Par conséquent, l'ensemble* O *est vide à la fin de la septième itération. L'algorithme se termine alors en retournant l'ensemble* S *contenant* ℓ_γ *et* ℓ'_γ*, la première étiquette représentant alors un chemin solution nécessairement optimal.*

En pratique, la recherche d'un chemin solution nécessairement optimal peut engendrer un nombre important de questions durant la résolution. Par conséquent, l'approche proposée dans cette sous-section se présente comme une réponse théorique à la problématique de recommandation d'un chemin dans un graphe d'états. Pour la mise en œuvre de cette approche, il convient de baisser nos exigences sur la qualité de la recommandation en nous autorisant un seuil de tolérance $\delta > 0$. L'objectif est alors de déterminer un vecteur solution *presque optimal*, i.e. un vecteur $x \in \mathcal{X}$ tel que $f_\omega(x) - f_\omega(y) \leq \delta$ pour tout

$\omega \in \Omega_{T+1}$ et tout $y \in \mathcal{X}$. Autrement dit, nous cherchons à identifier un vecteur solution $x \in \mathcal{X}$ tel que $\mathrm{MR}(x, \mathcal{X}, \Omega_{T+1}) \leq \delta$. Ceci peut par exemple être réalisé en modifiant les stratégies S1 et S2 de manière à poser des questions uniquement lorsque les valeurs mMR sont strictement plus grandes que δ. En effet, cela permet de détecter une étiquette $\ell' \in S$ telle que l'inégalité $\mathrm{MR}(g_{\ell'}, \{g_\ell : \ell \in S\}, \Omega_{T+1}) \leq \delta$ est vraie à la fin de la résolution. Par ailleurs, puisque nous nous autorisons à présent une erreur bornée par le seuil δ, nous devons aussi modifier les règles d'élagages R1 et R2 pour ne plus éliminer des étiquettes pouvant conduire à des chemins solutions presque optimaux. Ceci se fait tout simplement en remplaçant la relation $\prec_\Omega$ par sa version "approchée" $\prec_\Omega^\delta$ définie par : $Z \prec_\Omega^\delta y \Leftrightarrow \forall \omega \in \Omega, \exists z \in Z, f_\omega(y) - f_\omega(z) > \delta$.

3.1.4 Résultats expérimentaux

Dans cette sous-section, nous présentons des résultats expérimentaux permettant d'évaluer les performances de nos algorithmes de recherche interactifs. Nous considérons ici des instances de $G = (N, A)$ avec un seul nœud but γ, engendré comme suit : les nœuds dans N sont tirés aléatoirement dans la grille à deux dimensions $\{1, \ldots, 1000\} \times \{1, \ldots, 1000\}$ de manière uniforme, sauf le nœud γ et le nœud source s qui sont situés en $(1000, 500)$ et $(1, 500)$ respectivement. Puis, chaque nœud engendré est relié à ses cinq plus proches voisins par des arcs dont les évaluations sont tirées uniformément dans $\{0, \ldots, 100\}^q$. Comme évaluation heuristique, nous considérons uniquement le point idéal $I(n) = (I_1(n), \ldots, I_q(n))$ pour tout $n \in N$, où $I_j(n) = \min_{x \in \{g(p) : p \in P(n,\gamma)\}} x_j$ pour tout $j \in \mathcal{Q}$. La valeur $I_j(n)$ est obtenue au préalable en appliquant l'algorithme A* au graphe évalué uniquement sur le critère j.

Nous évaluons ici les performances de l'algorithme 4, combiné avec les stratégies S1 ou S2, en terme de temps de résolution [1] et nombre de questions engendrées. À titre de comparaison, nous considérons aussi la méthode en deux phases (nommée S0 ci-après) consistant à lancer tout d'abord MOA* puis appliquer la stratégie CSS sur l'ensemble des vecteurs coûts retournés (c'est-à-dire sur les vecteurs coût Pareto-optimaux). Les tests réalisés ont pour objectif d'évaluer l'impact de q le nombre de critères (cf. table 3.1) et celui de $|N|$ le nombre de nœuds du graphe (cf. table 3.2) sur les performances de ces algorithmes de résolution. Les réponses aux questions sont ici simulées par une somme pondérée f_ω dont le jeu de poids ω a été choisi aléatoirement dans $\Omega = \{\omega \in \mathrm{int}(\mathbb{R}^q_+) : \sum_{j \in \mathcal{Q}} \omega_j = 1\}$ où int représente l'intérieur du cône.

TABLE 3.1 – Impact du nombre critères q sur les temps de résolution ($|N| = 200$, $\delta = 0.1$).

	$q = 2$		$q = 3$		$q = 4$		$q = 5$	
	temps	questions	temps	questions	temps	questions	temps	questions
S0	0.0	3.7	1.3	8.3	131.5	12.5	192.7	23.5
S1	2.9	3.5	25.1	6.8	78.2	10.2	603.2	13.1
S2	0.5	6.3	1.1	12.5	1.5	17.9	2.2	28.1

1. Les temps de calcul sont donnés en minutes, pour des expériences réalisées sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire. Les tests de dominance sont effectués par le solveur Gurobi depuis un programme écrit en Java.

TABLE 3.2 – Impact du nombre de noeuds $|N|$ sur les temps de calcul ($q = 3$,$\delta = 0.01$).

	$\|N\| = 100$		$\|N\| = 300$		$\|N\| = 400$		$\|N\| = 500$	
	temps	questions	temps	questions	temps	questions	temps	questions
S0	0.2	7.7	6.0	9.4	15.8	10.4	28.8	11.5
S1	2.4	6.5	158.4	7.7	65.6	9.1	154.4	8.3
S2	0.3	12.3	8.3	13.2	3.8	11.6	5.8	10.6

En comparant les résultats des stratégies S0 et S1, nous voyons tout d'abord que S1 engendre beaucoup moins de questions que S0. Par exemple, la stratégie S1 pose environ 45% de questions en moins que S0 pour $q = 5$ (cf. table 3.1) et permet une économie d'environ 25% sur les plus grandes instances (cf. table 3.2). Ainsi, intégrer l'élicitation à la résolution permet ici de réduire de manière significative le nombre de questions posées au décideur. Cependant, les temps de calcul sont plus mauvais avec S1, et plus particulièrement sur les petites instances et avec peu de critères (deux ou trois). Comparons maintenant les performances des stratégies S0 et S2. La stratégie S2 est quant à elle meilleure que S0 en terme de temps de résolution à partir de trois critères (cf. table 3.1) ; en particulier, S2 va presque 100 fois plus vite que S0 pour $q = 4, 5$. Par ailleurs, nous voyons que S2 est aussi plus rapide que S0 sur les grandes instances (cf. table 3.2). Par exemple, S2 prend environ 5 fois moins de temps que S0 pour $|N| = 500$ (tout en posant environ 10% de questions en moins). De plus, il semblerait que le nombre de questions engendrées par S2 tend à devenir plus petit lorsque q augmente ; en effet, S2 engendre environ 70%, 50%, 40% et 20% de questions de plus que S0 pour $q = 2, 3, 4$ et 5 respectivement (cf. table 3.1).

Finalement, notre algorithme semble être plus performant lorsque celui-ci est combiné avec S2 plutôt que S1. Néanmoins, la stratégie S1 doit être privilégiée dans les situations où le nombre d'interactions possibles avec le décideur est particulièrement limité.

3.1.5 Des algorithmes pour l'utilité espérée à la Choquet

Nous nous intéressons à présent à une famille de fonctions f_ω permettant de modéliser des comportements plus complexes que les sommes pondérées, appelées les *utilités espérées à la Choquet* [Schmeidler, 1986]. Celles-ci associent à chaque vecteur coût $x \in \mathcal{X}$ la valeur $\mathrm{Ch}(u(x), v)$ (cf. définition 16), où $u(x) = (u_1(x), \ldots, u_q(x_q))$. Plus précisément, nous avons :

$$\mathrm{Ch}(u(x), v) = \sum_{j \in \mathcal{Q}} \Big[u_{(j)}(x_{(j)}) - u_{(j-1)}(x_{(j-1)})\Big] v(X_{(j)}), \text{ avec } x_{(0)} = 0$$

où (.) est une permutation des critères telle que $u_{(1)}(x_{(1)}) \leq \ldots \leq u_{(q)}(x_{(q)})$, les ensembles $X_{(j)}, j \in \mathcal{Q}$, sont les ensembles de niveaux associés (cf. définition 15) et v est une capacité (cf. définition 13). Pour tout $j \in \mathcal{Q}$, la fonction $u_j : \mathbb{R}_+ \to [0, 1]$ est une fonction de désutilité croissante mesurant le coût subjectif de la performance x_j sur le critère $j \in \mathcal{Q}$ pour le décideur. La famille des utilités espérées à la Choquet contient en particulier les *utilités additives* [Fishburn, 1968] qui s'obtiennent en prenant une capacité

v additive (cf. définition 14). Dans cette section, nous considérons que l'imprécision sur les préférences du décideur porte uniquement sur la capacité v ; nous supposons donc que les fonctions de désutilités mono-critères u_j ont été élicitées au préalable à l'aide de techniques classiques (e.g., [Keeney and Raiffa, 1976, Bana e Costa and Vansnick, 2000]). Ainsi, l'ensemble Ω représente ici l'ensemble des capacités normalisées v compatibles avec $\mathcal{P}$ l'ensemble des données de préférences disponibles. Observons que $\mathrm{Ch}(u(x), v)$ est une fonction linéaire en v bien que $\mathrm{Ch}(u(x), v)$ ne soit pas linéaire en x. Par conséquent, toute préférence du type "a est meilleur que b" induit une contrainte linéaire sur l'espace des capacités admissibles, à savoir $\mathrm{Ch}(u(a), v) \leq \mathrm{Ch}(u(b), v)$. De ce fait, l'ensemble Ω des capacités compatibles avec les préférences observées forme ici aussi un polyèdre convexe.

i) Contexte

La principale difficulté du passage d'une somme pondérée à une intégrale de Choquet provient du fait que la fonction d'agrégation n'est plus linéaire en x. Par exemple, $\mathrm{Ch}(u(x)+u(y), v) \neq \mathrm{Ch}(u(x), v)+\mathrm{Ch}(u(y), v)$ en général. De ce fait, $\mathrm{Ch}(u(x), v) \leq \mathrm{Ch}(u(y), v)$ n'implique pas toujours $\mathrm{Ch}(u(x)+u(z), v) \leq \mathrm{Ch}(u(y)+u(z), v)$, ce qui signifie que le principe de Bellman n'est pas vérifié par les intégrales de Choquet. Par conséquent, il n'est plus possible d'utiliser la fonction d'agrégation pour élaguer en cours de résolution des chemins par comparaison avec d'autres chemins (arrivant au même nœud $n \notin \Gamma$). En particulier, la règle d'élagage R1, présentée dans la Section 3.1.3, n'est plus valable ici. Il reste néanmoins possible de supprimer des chemins en comparant leurs évaluations heuristiques avec les vecteurs coût associés aux chemins solutions déjà découverts. En effet, si H est admissible, alors pour tout $n \in N$, pour tout $p \in P(s, n)$ et pour tout $p' \in P(n, \Gamma)$, il existe $h \in H(n)$ tel que $g(p) + h \precsim_P g(p \circ p')$, où $\circ$ représente l'opérateur de concaténation de chemin. Comme par ailleurs $\mathrm{Ch}(u(.), v)$ est un agrégateur monotone, alors nous en déduisons $\mathrm{Ch}(u(g(p) + h), v) \leq \mathrm{Ch}(u(g(p \circ p')), v)$. Ainsi, si $\mathrm{Ch}(u(g(p'')), v) \leq \mathrm{Ch}(u(g(p) + h), v)$ pour au moins un chemin solution découvert p'', alors $\mathrm{Ch}(u(g(p'')), v) \leq \mathrm{Ch}(u(g(p \circ p')), v)$ par transitivité. Dans ce cas, le chemin solution p'' est préféré au chemin solution $p \circ p'$ (qui est issu du chemin p). Par conséquent, lorsque v est imprécisément connu, nous pouvons utiliser la règle d'élagage R2 pour éliminer des chemins en cours de résolution par comparaison avec des chemins solutions découverts. Soulignons que les tests de dominance engendrés par cette règle peuvent ici aussi être réalisés par programmation linéaire (cf. proposition 18) car l'intégrale de Choquet est une fonction linéaire en ses paramètres v. Cependant, dans nos expériences, nous verrons que la règle R2 ne permet pas à elle seule de produire un algorithme de recherche efficace. C'est pourquoi nous attaquons maintenant le problème sous l'angle d'une recherche approchée.

Pour les problèmes de recherche dans un graphe d'états, la recherche approchée avec garantie de performance a tout d'abord été étudiée dans le cadre d'une évaluation mono-critère ; en particulier, plusieurs variantes de l'algorithme A* ont été proposées dans la littérature (e.g., [Pearl and Kim, 1982, Ghallab and Allard, 1983]). Ces algorithmes de résolution garantissent de déterminer une solution dont le coût n'excède pas la valeur optimale d'un facteur plus grand que $(1 + \varepsilon)$ où $\varepsilon > 0$. L'objectif de ces algorithmes est de parvenir à un compromis raisonnable entre la garantie d'optimalité de la solution

retournée et les temps de calcul. Ces variantes se sont montrées particulièrement efficaces en pratique, réalisant par exemple une économie, en terme de nombre de nœuds développés, atteignant 90% sur certaines instances avec $\varepsilon = 0.1$ (e.g., [Pearl and Kim, 1982, Ghallab and Allard, 1983]).

En optimisation multicritère, la recherche approchée présente un autre avantage pratique majeur qui est celui de réduire significativement le nombre de solutions retournées. En effet, il ne s'agit plus de déterminer l'ensemble $\mathrm{ND}(\mathcal{X})$ des vecteurs coût Pareto-optimaux, mais de trouver une approximation à $(1+\varepsilon)$-près de cet ensemble :

Définition 49 ($(1+\varepsilon)$-approximation)**.** *L'ensemble $X \subset \mathbb{R}^q_+$ est une approximation à $(1+\varepsilon)$-près de l'ensemble $Y \subset \mathbb{R}^q_+$ si et seulement si : $\forall y \in Y,\ \exists x \in X,\ \forall j \in \mathcal{Q},\ x_j \leq (1+\varepsilon)y_j$. Dans ce cas, X est aussi appelée une $(1+\varepsilon)$-approximation de l'ensemble Y.*

Le résultat suivant permet de justifier l'engouement actuel pour la recherche approchée :

Proposition 24 ([Papadimitriou and Yannakakis, 2000])**.** *Pour tout problème d'optimisation multicritère et tout réel $\varepsilon > 0$, il existe une $(1+\varepsilon)$-approximation des solutions réalisables de taille polynomiale en la taille de l'instance et en $1/\varepsilon$.*

Ainsi, même si l'ensemble $\mathrm{ND}(\mathcal{X})$ est de taille exponentielle, il existe une $(1+\varepsilon)$-approximation des solutions réalisables de taille polynomiale en la taille de l'instance et en $1/\varepsilon$. Un algorithme de recherche approchée est alors qualifié de *Fully Polynomial Time Approximation Scheme* (FPTAS) si celui-ci a une complexité en temps et en espace qui est polynomiale en la taille de l'instance et en $1/\varepsilon$; en supposant que P $\neq$ NP, un FPTAS est le mieux que nous puissions espérer pour un problème NP-difficile.

Dans la littérature, plusieurs FPTAS ont été proposés pour le calcul d'une $(1+\varepsilon)$-approximation dans le cadre de problèmes de plus court chemins multicritères (qui est un cas particulier de la recherche dans un graphe d'états multicritère). À titre d'exemple, nous pouvons citer les travaux de [Hansen, 1980, Ergun et al., 2002] pour le cas particulier du bicritère, ceux de [Warburton, 1987] pour les graphes acycliques, ceux de [Perny and Spanjaard, 2008] pour les graphes d'états disposant d'une borne supérieure sur la taille maximale des chemins, ou encore ceux de [Tsaggouris and Zaroliagis, 2009a] pour une généralisation de l'algorithme de Bellman-Ford en mono-critère. Tous ces algorithmes reposent sur la programmation dynamique, ce qui semble malheureusement exclure la possibilité de les étendre à des modèles non linéaires comme l'utilité espérée à la Choquet.

Pour contourner cette difficulté, des règles d'élagage spécifiques à l'intégrale de Choquet ont récemment été proposées, permettant de comparer des chemins arrivant dans un même nœud en utilisant la fonction d'agrégation (e.g., [Galand et al., 2013]). Cependant, après quelques expériences, nous nous sommes rendu compte que l'extension de ces règles au cas de capacités v imprécisément connues ne permet pas de réduire le nombre de nœuds développés de manière significative. Un autre angle d'attaque consisterait à abandonner l'idée de comparer des sous-chemins entre eux pour améliorer les règles qui permettent de supprimer des chemins par comparaison avec des chemins solutions déjà trouvés. Cette voie a été explorée relativement récemment, conduisant à la proposition d'une variante de l'algorithme MOA* nommée MOA*$_\varepsilon$: une étiquette $\ell = [n_\ell, p_\ell, g_\ell]$ est éliminée si, pour tout vecteur heuristique $h \in H(n_\ell)$,

il existe un chemin solution découvert p tel que $g_\ell + h \precsim_P (1+\varepsilon)g(p)$ [Perny and Spanjaard, 2008]. Ce nouvel algorithme est une généralisation de l'algorithme MOA* (quand $\varepsilon = 0$, MOA*$_\varepsilon$ et MOA* sont équivalents) permettant de retourner une $(1+\varepsilon)$-approximation de l'ensemble des vecteurs coût Pareto-optimaux. Par ailleurs, les expériences menées par les auteurs ont montré que MOA*$_\varepsilon$ est très efficace en pratique, bien que ce ne soit pas un FPTAS. C'est pourquoi nous proposons maintenant une adaptation de cet algorithme au cas de préférences représentables par une utilité espérée à la Choquet imprécisément connue.

ii) Calcul d'une $(1+\varepsilon)$-approximation des solutions réalisables

Dans cette section, nous supposons que la capacité de Choquet v est imprécisément connue et que le décideur ne peut pas répondre à nos éventuelles questions servant à réduire l'ensemble Ω des capacités admissibles. Dans ce contexte, notre objectif est de déterminer un ensemble de chemins solutions qui, pour toutes les capacités admissibles $v \in \Omega$, contient une solution optimale au sens de $\mathrm{Ch}(u(.), v)$ à un facteur $(1+\varepsilon)$-près. Pour pouvoir caractériser formellement un tel ensemble, nous commençons par introduire la relation de dominance suivante :

Définition 50 ((ε,Ω)-dominance)**.** *Pour tout $\varepsilon \geq 0$, tout $y \in \mathbb{R}^q_+$ et tout $Z \subseteq \mathbb{R}^q_+$:*

$$Z \precsim^\varepsilon_\Omega y \Leftrightarrow \forall v \in \Omega, \exists z \in Z, \mathrm{Ch}(u(z), v) \leq (1+\varepsilon)\,\mathrm{Ch}(u(y), v)$$

Ainsi, nous souhaitons identifier un ensemble de vecteurs solutions $X \subseteq \mathcal{X}$ tel que $X \precsim^\varepsilon_\Omega x$ pour tout vecteur solution $x \in \mathcal{X}$. Dans la suite, un tel ensemble sera appelé une (ε,Ω)-*approximation* de l'ensemble $\mathcal{X}$. En pratique, il convient de déterminer une (ε,Ω)-approximation de l'ensemble $\mathcal{X}$ de taille suffisamment petite pour que le décideur puisse facilement faire un choix parmi ces éléments. Observons que l'ensemble $\mathrm{ND}(\mathcal{X})$ est trivialement une (ε,Ω)-approximation de l'ensemble $\mathcal{X}$ car $\mathrm{Ch}(u(.), v)$ est monotone, mais que cette ensemble peut contenir un nombre exponentiel d'éléments (e.g., [Hansen, 1980]). Néanmoins, cette observation montre que l'algorithme MOA* constitue (une nouvelle fois) une base intéressante pour déterminer une (ε,Ω)-approximation de l'ensemble $\mathcal{X}$ de taille plus petite. Afin de limiter la taille de l'approximation construite, il convient de définir de nouvelles règles d'élagage permettant de repérer rapidement des étiquettes menant à des vecteurs solutions $\precsim^\varepsilon_\Omega$-dominés par l'ensemble des vecteurs coût associés aux chemins solutions déjà découverts (i.e. $\{g_\ell : \ell \in S \bigcup_{\gamma\in\Gamma} O(\gamma)\}$). Pour ce faire, nous exploitons l'heuristique H de MOA* car celle-ci permet de garantir le résultat suivant (quand H est admissible) :

Proposition 25. *Pour toute étiquette $\ell' \in O$, si $\{g_\ell : \ell \in S \bigcup_{\gamma\in\Gamma} O(\gamma)\} \precsim^\varepsilon_\Omega f'$ pour tout $f' \in F(\ell')$, alors tout vecteur solution issu de $p_{\ell'}$ est $\precsim^\varepsilon_\Omega$-dominé par les vecteurs solutions $\{g_\ell : \ell \in S \bigcup_{\gamma\in\Gamma} O(\gamma)\}$.*

Démonstration. Posons $L = S \bigcup_{\gamma\in\Gamma} O(\gamma)$. Soit $\ell' \in O$ telle que $\{g_\ell : \ell \in L\} \precsim^\varepsilon_\Omega f'$ pour tout $f' \in F(\ell')$. Montrons que nous avons $\{g_\ell : \ell \in L\} \precsim^\varepsilon_\Omega g(p_{\ell'} \circ p)$ pour tout chemin $p \in P(n_{\ell'}, \Gamma)$, où $\circ$ représente l'opérateur de concaténation de chemins. Tout d'abord, puisque H est admissible, alors il existe $h \in H(n_{\ell'})$ tel que $h \precsim_P g(p)$, ce qui implique $h + g_{\ell'} \precsim_P g(p) + g_{\ell'}$. Ainsi, il existe $f' \in F(\ell')$

tel que $f' \precsim_P g(p) + g_{\ell'} = g(p_{\ell'} \circ p)$. Comme les fonctions $u_j, j \in \mathcal{Q}$, sont croissantes, alors nous avons $u(f') \leq u(g(p_{\ell'} \circ p))$. Par ailleurs, puisque l'intégrale de Choquet est monotone, alors nous en déduisons $\mathrm{Ch}(u(f'), v) \leq \mathrm{Ch}(u(g(p_{\ell'} \circ p)), v)$. Ensuite, comme $\{g_\ell : \ell \in L\} \precsim^{\varepsilon}_{\Omega} \{g_{\ell'}\} + H(n)$ par hypothèse, alors pour tout $v \in \Omega$, il existe une étiquette $\ell \in L$ telle que $\mathrm{Ch}(u(g_\ell), v) \leq (1+\varepsilon)\,\mathrm{Ch}(u(f'), v)$. Par conséquent, nous obtenons $\mathrm{Ch}(u(g_\ell), v) \leq (1+\varepsilon)\,\mathrm{Ch}(u(f'), v) \leq (1+\varepsilon)\,\mathrm{Ch}(u(g(p_{\ell'} \circ p)), v)$. Ainsi, le vecteur solution $g(p_{\ell'} \circ p)$ est $\precsim^{\varepsilon}_{\Omega}$-dominé par les vecteurs solutions $\{g_\ell : \ell \in S \bigcup_{\gamma \in \Gamma} O(\gamma)\}$. □

Cette proposition nous autorise donc à éliminer des chemins par comparaison avec des chemins solutions découverts en utilisant la règle d'élagage suivante :

Règle R3. *Supprimer l'étiquette $\ell' \in O$ si $\{g_\ell : \ell \in S \bigcup_{\gamma \in \Gamma} O(\gamma)\} \precsim^{\varepsilon}_{\Omega} f'$ pour tout $f' \in F(\ell')$.*

Soulignons que, tout comme la Ω-dominance, la (ε, Ω)-dominance peut être testée efficacement avec un solveur de programmes linéaires, grâce à l'équivalence suivante :

Proposition 26. *Pour tous ensembles $Y, Z \subseteq \mathbb{R}^q_+$:*

$$Z \precsim^{\varepsilon}_{\Omega} Y \;\Leftrightarrow\; \left[\forall y \in Y, \max_{v \in \Omega} \min_{z \in Z} \left\{ \mathrm{Ch}(u(z), v) - (1+\varepsilon)\,\mathrm{Ch}(u(y), v) \right\} \leq 0\right]$$

La preuve de cette proposition est délibérément omise car celle-ci est très similaire à la démonstration de la proposition 18. Finalement, l'adaptation de MOA$^*{}_{\varepsilon}$ que nous proposons pour les utilités espérées à la Choquet est résumée dans l'algorithme 5, où $\circ$ représente l'opérateur de concaténation de chemins.

Algorithm 5:

Input: $G = (N, A)$; s ; Γ ; H ; Ω
$O(s) \leftarrow \{[s, \langle s \rangle, (0, \ldots, 0)]\}$
$C,\ S \leftarrow \emptyset$
while $O \neq \emptyset$ **do**
  $\ell^* \leftarrow$ Choisir(O)
  Déplacer l'étiquette ℓ^* de l'ensemble des ouverts O vers l'ensemble des fermés C
  if $n_{\ell^*} \in \Gamma$ **then**
    Ajouter l'étiquette ℓ^* à l'ensemble de solutions S
  else
    foreach $n \in \Pi(n_{\ell^*})$ **do**
      Générer l'étiquette $\ell' = [n, p_{\ell^*} \circ \langle n \rangle, g_{\ell^*} + g(n_{\ell^*}, n)]$
      if *pour tout* $\ell \in O(n) \cup C(n)$, $\neg(g_\ell \precsim_P g_{\ell'})$ **then**
        Ajouter l'étiquette ℓ' à l'ensemble des ouverts O
        Appliquer la règle d'élagage R3 à l'étiquette ℓ'
      end
    end
  end
end
return une étiquette $\ell \in S$ par élément de l'ensemble $\{g_{\ell'} : \ell' \in S\}$

La validité de notre algorithme est établie par la proposition ci-dessous :

Proposition 27. *L'algorithme 5 retourne une* (ε, Ω)*-approximation de l'ensemble* $\mathcal{X}$*.*

Démonstration. Comme nous l'avons déjà remarqué, l'algorithme MOA* permet de retourner une (ε, Ω)-approximation de l'ensemble $\mathcal{X}$. Celui-ci est ici raffiné en utilisant la règle d'élagage R3. La proposition 25 nous assure que cette règle n'élimine que des étiquettes conduisant à des vecteurs solutions $\precsim^{\varepsilon}_{\Omega}$-dominées par des vecteurs solutions découverts par l'algorithme. Notons alors que tous les vecteurs solutions qui ont été utilisés par la règle R3 sont forcément dans l'ensemble retourné par l'algorithme car ces vecteurs sont ajoutés à l'ensemble S lors de leur développement (cf. ligne 7). Par conséquent, l'ensemble S est une (ε, Ω)-approximation de l'ensemble $\mathcal{X}$. □

L'algorithme 5 permet de calculer une (ε, Ω)-approximation de l'ensemble $\mathcal{X}$ par un algorithme de développement d'étiquettes. Cependant, quand l'ensemble Ω des capacités v possibles est très grand, le nombre minimal d'éléments dans une (ε, Ω)-approximation de $\mathcal{X}$ peut être bien trop important pour permettre au décideur de prendre une décision facilement. Dans ces situations, il s'agit de collecter des informations sur les préférences du décideur pour pouvoir formuler une recommandation personnalisée.

iii) Un algorithme interactif pour déterminer une solution optimale à $(1+\varepsilon)$-près

Lorsque le décideur est disponible pour répondre à nos questions, nous pouvons envisager de l'interroger pour déterminer un vecteur solution *nécessairement optimal à un facteur* $(1+\varepsilon)$*-près* : un vecteur solution $x \in \mathcal{X}$ est dit nécessairement optimal à un facteur $(1+\varepsilon)$-près si et seulement si $\mathrm{Ch}(u(x), v) \leq (1+\varepsilon)\,\mathrm{Ch}(u(y), v)$ pour tout $y \in \mathcal{X}$ et pour tout $v \in \Omega$. À la fin de l'exécution de l'algorithme 5, s'il existe une étiquette $\ell^* \in S$ telle que $\mathrm{Ch}(u(g_{\ell^*}), v) \leq \mathrm{Ch}(u(g_\ell), v)$ pour toute étiquette $\ell \in S$ et pour toute capacité $v \in \Omega$, alors le vecteur coût g_{ℓ^*} est en fait nécessairement optimal à un facteur $(1+\varepsilon)$-près. En effet, comme l'ensemble $\{g_\ell : \ell \in S\}$ forme une $(1+\varepsilon)$-approximation de $\mathcal{X}$ (cf. proposition 27), alors pour tout vecteur solution $y \in \mathcal{X}$ et toute capacité $v \in \Omega$, il existe une étiquette $\ell \in S$ telle que $\mathrm{Ch}(u(g_\ell), v) \leq (1+\varepsilon)\,\mathrm{Ch}(u(y), v)$, ce qui implique $\mathrm{Ch}(u(g_{\ell^*}), v) \leq (1+\varepsilon)\,\mathrm{Ch}(u(y), v)$ par transitivité. De ce fait, s'il existe une étiquette $\ell^* \in S$ telle que $\mathrm{MR}(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega) \leq 0$ à la fin de son exécution, alors le vecteur coût g_{ℓ^*} est nécessairement optimal à un facteur $(1+\varepsilon)$-près. Dans le cas contraire, on pourrait envisager de poser des questions au décideur après son exécution tant que $\mathrm{MR}(g_{\ell'}, \{g_\ell : \ell \in S\}, \Omega) > 0$ pour toute étiquette $\ell' \in S$ (autrement dit tant que $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega) > 0$). Néanmoins, cette approche ne serait pas très efficace en pratique car l'ensemble S peut être très grand si l'ensemble Ω initial n'est pas suffisamment discriminant. C'est pourquoi nous proposons à la place d'intégrer l'élicitation à la résolution pour réduire les temps de résolution tout en garantissant de retourner un vecteur solution nécessairement optimal à un facteur $(1+\varepsilon)$-près. Poser des questions en cours de résolution est envisageable car nous avons la propriété suivante :

$$\forall \Omega' \subseteq \Omega, \forall y \in \mathbb{R}^q_+, \forall Z \subset \mathbb{R}^q_+,\ Z \precsim^{\varepsilon}_{\Omega} y \Rightarrow Z \precsim^{\varepsilon}_{\Omega'} y \tag{3.1}$$

Ainsi, si l'ensemble Ω réduit en cours de résolution, alors nous sommes sûrs que les suppressions réalisées par la règle R3 ne seront pas remises en cause plus tard durant la recherche : l'ensemble retourné sera une (ε, Ω_f)-approximation de l'ensemble $\mathcal{X}$, où Ω_f représente l'ensemble des capacités admissibles à la fin de la résolution. À présent, il s'agit de déterminer à quels moments le décideur doit être sollicité pour garantir l'existence d'une étiquette $\ell^* \in S$ telle que $\mathrm{MR}(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega_f) \leq 0$. Dans l'optique de réduire le nombre de questions posées, nous pouvons par exemple utiliser la stratégie d'élicitation S1 introduite dans la Section 3.1.3. Rappelons que cette stratégie consiste à poser des questions dès que la valeur $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega)$ devient strictement positive (suite à l'insertion d'une étiquette dans S) et tant que cette valeur n'est pas devenue négative. En procédant de cette manière, l'étiquette $\ell^* \in S$ vérifiant $\mathrm{MR}(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega_f) \leq 0$ à la fin de l'exécution correspond par définition à un chemin solution nécessairement optimal à un facteur $(1+\varepsilon)$-près.

Expérimentalement, nous avons observé que cette procédure, mêlant élicitation et résolution, permet de réduire de manière significative les temps de calcul (par rapport à sa version sans élicitation). Cependant, exiger que la valeur $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega)$ soit inférieure à zéro en permanence peut engendrer un questionnaire bien trop long pour que le décideur accepte d'y prendre part. Dans le cadre de préférences représentables par une somme pondérée, nous avons pu contourner cette difficulté en nous autorisant un seuil de tolérance $\delta > 0$ sur la valeur mMR. En effet, dans nos expériences, nous avons vu que cela permettait d'obtenir un bon compromis entre le temps de résolution et le nombre de questions engendrées. Cependant, au sein de l'algorithme 5, une telle relaxation revient en quelque sorte à mélanger la garantie de performance à un facteur $(1+\varepsilon)$-près avec une erreur δ définie en terme de regrets (ou écart relatif), ce qui n'a pas vraiment de sens. Dans la sous-section suivante, nous allons montrer qu'il est néanmoins possible de restaurer une garantie de performance, à condition de combiner l'algorithme 5 avec une stratégie d'élicitation fondée sur un nouveau type de regrets.

iv) Un algorithme interactif pour trouver une solution de Max Regret inférieur à δ

Dans cette sous-section, nous proposons une méthode interactive permettant de déterminer un vecteur solution $x \in \mathcal{X}$ tel que $\mathrm{MR}(x, \mathcal{X}, \Omega) \leq \delta$, où $\delta > 0$ est un seuil de tolérance que le décideur juge acceptable. Cette méthode utilise l'algorithme 5 pour explorer l'espace des solutions de manière efficace, et corrige les erreurs commises par cet algorithme en considérant une stratégie d'élicitation fondée sur un nouveau type de regrets. Plus précisément, nous proposons de travailler avec les définitions de regrets suivantes :

Définition 51. *Pour tous $x, y \in \mathcal{X}$:*

$$\begin{aligned}
\mathrm{PMR}_\varepsilon(x, y, \Omega) &= \max_{v \in \Omega} \{(1+\varepsilon)\,\mathrm{Ch}(u(x), v) - \mathrm{Ch}(u(y), v)\} \\
\mathrm{MR}_\varepsilon(x, \mathcal{X}, \Omega) &= \max_{y \in \mathcal{X}} \mathrm{PMR}_\varepsilon(x, y, \Omega) \\
\mathrm{mMR}_\varepsilon(\mathcal{X}, \Omega) &= \min_{x \in \mathcal{X}} \mathrm{MR}_\varepsilon(x, \mathcal{X}, \Omega)
\end{aligned}$$

Ces définitions généralisent les concepts de regrets classiques qui se retrouvent tout simplement en prenant $\varepsilon = 0$. Ces nouveaux regrets permettent de corriger les erreurs faites par la règle d'élagage R3 grâce au résultat suivant :

Proposition 28. *Si* $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq (1+\varepsilon)\delta$ *à la fin de l'exécution de l'algorithme 5, alors il existe une étiquette* $\ell^* \in S$ *telle que* $\mathrm{MR}(g_{\ell^*}, \mathcal{X}, \Omega) \leq \delta$.

Démonstration. Posons $\lambda = (1+\varepsilon)\delta$. Supposons que nous avons $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq \lambda$ à la fin de l'exécution. Soit $\ell^* \in S$ telle que $\mathrm{MR}_\varepsilon(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega) = \mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega)$. Montrons que nous avons $\mathrm{MR}(g_{\ell^*}, \mathcal{X}, \Omega) \leq \delta$. Comme les fonctions $u_j, j \in \mathcal{Q}$, sont croissantes et que l'intégrale de Choquet est monotone, alors cela revient à montrer que $\mathrm{MR}(g_{\ell^*}, \mathrm{ND}(\mathcal{X}), \Omega) \leq \delta$. Pour ce faire, il suffit de prouver que $\mathrm{PMR}(g_{\ell^*}, x, \Omega) \leq \delta$ pour tout $x \in \mathrm{ND}(\mathcal{X})$. Soit $x \in \mathrm{ND}(\mathcal{X})$ un vecteur solution Pareto-optimal. Deux cas peuvent se produire :

- *Cas 1 : il existe* $\ell \in S$ *telle que* $g_\ell = x$. Puisque $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq \lambda$ par hypothèse et que $\mathrm{MR}_\varepsilon(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega) = \mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega)$ par définition de ℓ^*, alors nous avons :

$$\mathrm{PMR}_\varepsilon(g_{\ell^*}, x, \Omega) \leq \mathrm{MR}_\varepsilon(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega) = \mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq \lambda$$

  Par définition de PMR_ε, nous avons donc $(1+\varepsilon)\,\mathrm{Ch}(u(g_{\ell^*}), v) - \mathrm{Ch}(u(x), v) \leq \lambda$ pour tout $v \in \Omega$. Comme par ailleurs $\varepsilon \geq 0$, alors nous en déduisons $(1+\varepsilon)\,\mathrm{Ch}(u(g_{\ell^*}), v) - (1+\varepsilon)\,\mathrm{Ch}(u(x), v) \leq \lambda$, ce qui est équivalent à $\mathrm{Ch}(u(g_{\ell^*}), v) - \mathrm{Ch}(u(x), v) \leq \lambda/(1+\varepsilon) = \delta$. Ainsi, nous avons bien $\mathrm{PMR}(g_{\ell^*}, x, \Omega) \leq \delta$ par définition de PMR.
- *Cas 2 : pour tout chemin solution* p *de vecteur coût* x*, il existe une étiquette* ℓ' *telle que* $p_{\ell'}$ *est un sous-chemin de* p *et* ℓ' *est éliminée par la règle d'élagage R3.* Par définition de cette règle, pour tout $h \in H(n_{\ell'})$, nous avons $\{g_\ell : \ell \in S' \cup \bigcup_{\gamma\in\Gamma} O'(\gamma)\} \precsim^\varepsilon_\Omega g_{\ell'} + h$ où S' et $O'(\gamma)$ représentent respectivement les ensembles S et $O(\gamma)$ au moment de la suppression de l'étiquette ℓ'. Autrement dit, pour tout $h \in H(n_{\ell'})$ et pour tout $v \in \Omega$, il existe une étiquette $\ell'' \in S' \cup \bigcup_{\gamma\in\Gamma} O'(\gamma)$ telle que $(1+\varepsilon)\,\mathrm{Ch}(u(g_{\ell'} + h), v) \geq \mathrm{Ch}(u(g_{\ell''}), v)$. Par ailleurs, comme l'heuristique H est admissible, alors il existe $h' \in H(n_{\ell'})$ tel que $g_{\ell'} + h' \precsim_P x$. Par conséquent, nous avons $\mathrm{Ch}(u(g_{\ell'} + h'), v) \leq \mathrm{Ch}(u(x), v)$ car $u_j, j \in \mathcal{Q}$, est croissante et l'intégrale de Choquet est monotone. Des deux inégalités précédentes, nous déduisons $(1+\varepsilon)\,\mathrm{Ch}(u(x), v) \geq \mathrm{Ch}(u(g_{\ell''}), v)$. De plus, comme $S' \cup \bigcup_{\gamma\in\Gamma} O'(\gamma) \subseteq S$ par construction et $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq \lambda$ par hypothèse, alors nous avons $(1+\varepsilon)\,\mathrm{Ch}(u(g_{\ell^*}), v) - \mathrm{Ch}(u(g_{\ell''}), v) \leq \lambda$. Ainsi, nous déduisons $(1+\varepsilon)\,\mathrm{Ch}(u(g_{\ell^*}), v) - (1+\varepsilon)\,\mathrm{Ch}(u(x), v) \leq \lambda$ des deux dernières inégalités, ce qui se réécrit $\mathrm{Ch}(u(g_{\ell^*}), v) - \mathrm{Ch}(u(x), v) \leq \lambda/(1+\varepsilon) = \delta$. D'où $\mathrm{PMR}(g_{\ell^*}, x, \Omega) \leq \delta$.

Finalement, nous venons de montrer que toute étiquette $\ell^* \in S$ vérifiant $\mathrm{MR}_\varepsilon(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega) = \mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega)$ est telle que $\mathrm{MR}(g_{\ell^*}, \mathcal{X}, \Omega) \leq \delta$. □

Ainsi, si la valeur $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega)$ est inférieure à la valeur $(1+\varepsilon)\delta$ à la fin de la recherche, alors l'algorithme 5 retourne au moins une étiquette $\ell^* \in S$ vérifiant $\mathrm{MR}(g_{\ell^*}, \mathcal{X}, \Omega) \leq \delta$. Dans le cas contraire, nous pouvons envisager de poser des questions au décideur de sorte à réduire l'ensemble Ω tant que $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) > (1+\varepsilon)\delta$. Cependant, pour ce nouveau type de regrets, construire un questionnaire garantissant que nous ayons $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq (1+\varepsilon)\delta$ à un moment donné n'est pas aussi évident :

Proposition 29. *Si $\varepsilon \leq \delta/(1-\delta)$, alors il existe un questionnaire permettant de garantir que la condition $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq (1+\varepsilon)\delta$ est satisfaite après un nombre fini de questions.*

Démonstration. Considérons le graphe orienté $G' = (N', A')$ où :

- N' est l'ensemble des nœuds, composé d'un nœud n'_ℓ par étiquette $\ell \in S$.
- A' est l'ensemble des arcs de ce graphe, un nœud $n'_{\ell_1} \in N'$ étant relié au nœud $n'_{\ell_2} \in N'$ si et seulement si $\mathrm{Ch}(u(g_{\ell_1}), v) \leq \mathrm{Ch}(u(g_{\ell_2}), v)$ pour tout $v \in \Omega$.

Pour tout circuit de type $\langle n'_{\ell_1}, \ldots, n'_{\ell_k}, n'_{\ell_1} \rangle$ dans G', nous avons donc $\mathrm{Ch}(u(g_{\ell_1}), v) = \ldots = \mathrm{Ch}(u(g_{\ell_k}), v)$ pour tout $v \in \Omega$. Par conséquent, considérons à la place le graphe orienté sans circuit $G'' = (N'', A'')$ obtenu à partir de $G' = (N', A')$ en remplaçant chaque circuit maximal par un unique nœud représentatif. Ce graphe est ici utilisé pour définir une stratégie d'élicitation permettant d'assurer que la condition $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq (1+\varepsilon)\delta$ est satisfaite après un nombre fini de questions. Présentons maintenant cette stratégie. Tant que $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) > (1+\varepsilon)\delta$ et que le graphe G'' contient au moins deux nœuds sans parent :

1. Déterminer un nœud $n''_{\ell_1} \in N''$ tel que $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega) = \mathrm{MR}(g_{\ell_1}, \{g_\ell : \ell \in S\}, \Omega)$. Soit $n''_{\ell^\top}$ un ancêtre de n''_{ℓ_1} qui n'a pas de parent dans G''. Si n''_{ℓ_1} n'a pas d'ancêtre, alors nous prenons $n''_{\ell^\top} = n''_{\ell_1}$. Par définition du graphe G'', nous avons $\mathrm{Ch}(u(g_{\ell^\top}), v) \leq \mathrm{Ch}(u(g_{\ell_1}), v)$ pour tout $v \in \Omega$. D'où $\mathrm{mMR}(\{g_\ell : \ell \in S\}, \Omega) = \mathrm{MR}(g_{\ell_1}, \{g_\ell : \ell \in S\}, \Omega) = \mathrm{MR}(g_{\ell^\top}, \{g_\ell : \ell \in S\}, \Omega)$.
2. Déterminer un nœud $n''_{\ell_2} \in N''$ tel que $\mathrm{MR}(g_{\ell^\top}, \{g_\ell : \ell \in S\}, \Omega) = \mathrm{PMR}(g_{\ell^\top}, g_{\ell_2}, \Omega)$. Soit $n''_{\ell^\perp}$ un ancêtre de n''_{ℓ_2} qui n'a pas de parent dans G''. Si n''_{ℓ_2} n'a pas d'ancêtre, alors nous prenons ici aussi $n''_{\ell^\perp} = n''_{\ell_2}$. Notons que nous avons forcément $n''_{\ell^\top} \neq n''_{\ell^\perp}$ car sinon $n''_{\ell^\top}$ serait ancêtre de tous les autres nœuds du graphe G'' ; en effet, on aurait dans ce cas $\mathrm{MR}(g_{\ell^\top}, \{g_\ell : \ell \in S\}, \Omega) = \mathrm{PMR}(g_{\ell^\top}, g_{\ell_2}, \Omega) = \mathrm{PMR}(g_{\ell^\top}, g_{\ell^\top}, \Omega) = 0$.
3. Demander au décideur de comparer les chemins solutions associés aux étiquettes $\ell^\top$ et $\ell^\perp$.
4. Mettre à jour le graphe G' en y ajoutant tous les arcs induits par la réponse du décideur, puis modifier le graphe G'' en fonction.

À la sortie de la boucle "tant que", nous avons soit $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq (1+\varepsilon)\delta$ (et dans ce cas, il n'y a plus rien à démontrer) soit le graphe G'' ne possède plus qu'un seul nœud n''_{ℓ^*} sans parent. Soulignons que ce nœud est alors ancêtre de tous les autres nœuds du graphe G''. De ce fait, pour tout $\ell \in S$, nous avons $\mathrm{Ch}(u(g_{\ell^*}), v) \leq \mathrm{Ch}(u(g_\ell), v)$ pour tout $v \in \Omega$ (par définition de G'') et par conséquent :

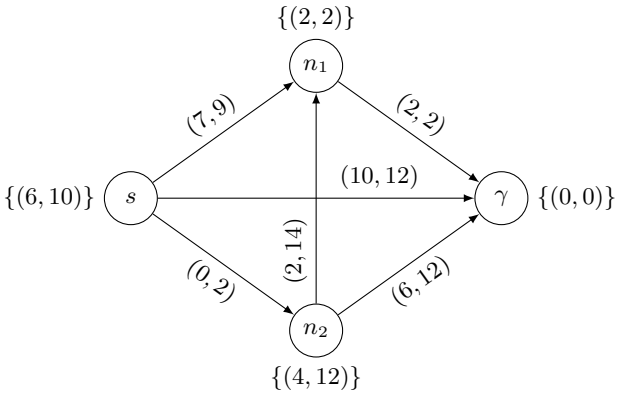
$$
\begin{aligned}
\mathrm{PMR}_\varepsilon(g_{\ell^*}, g_\ell, \Omega) &= \max_{v \in \Omega} \{(1+\varepsilon)\,\mathrm{Ch}(u(g_{\ell^*}), v) - \mathrm{Ch}(u(g_\ell), v)\} \quad \text{par définition de } \mathrm{PMR}_\varepsilon \\
&= \max_{v \in \Omega} \{(1+\varepsilon)\,\mathrm{Ch}(u(g_{\ell^*}), v) - \mathrm{Ch}(u(g_{\ell^*}), v)\} \quad \text{car } \forall v \in \Omega, \mathrm{Ch}(u(g_{\ell^*}), v) \leq \mathrm{Ch}(u(g_\ell), v) \\
&= \varepsilon \times \max_{v \in \Omega} \mathrm{Ch}(u(g_{\ell^*}), v) \\
&\leq \varepsilon \quad \text{car } \forall x \in [0,1]^q, \forall v \in \Omega, \mathrm{Ch}(x, v) \leq 1 \\
&\leq (1+\varepsilon)\delta \quad \text{car } \varepsilon \leq \delta/(1-\delta) \text{ par hypothèse}
\end{aligned}
$$

Ainsi, à la fin de la bouche "tant que", nous avons $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq \mathrm{MR}_\varepsilon(g_{\ell^*}, \{g_\ell : \ell \in S\}, \Omega) = \max_{\ell \in S} \mathrm{PMR}_\varepsilon(g_{\ell^*}, g_\ell, \Omega) \leq (1+\varepsilon)\delta$. □

Cette proposition nous informe que, si $\varepsilon \leq \delta/(1-\delta)$, alors la condition $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) \leq (1+\varepsilon)\delta$ peut être satisfaite en choisissant bien les questions à poser au décideur à la fin de l'exécution de l'algorithme 5. Par ailleurs, la preuve de cette proposition nous donne une stratégie d'élicitation permettant de parvenir à ce résultat en posant au plus $|S| - 1$ questions. Néanmoins, il semble plus judicieux d'intégrer l'élicitation à la résolution pour pourvoir réduire les temps de calcul ainsi que le nombre de questions posées (rappelons que cette approche est rendue possible par l'équation (3.1)). C'est pourquoi, nous proposons de combiner l'algorithme 5 à la stratégie d'élicitation suivante :

Stratégie S3. Cette stratégie consiste à poser des questions au décideur uniquement lorsque la valeur $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega)$ devient strictement supérieure à la valeur $(1+\varepsilon)\delta$ suite à l'insertion d'une nouvelle solution dans S. Dans cette situation, S3 interroge le décideur tant que $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) > (1+\varepsilon)\delta$. Par exemple, les questions peuvent être engendrées en suivant la stratégie décrite dans la preuve de la proposition 29.

Exemple 25. *Déroulons l'algorithme 5 combiné avec S3 sur le graphe bicritère $G = (N, A)$ suivant :*



Sur cette figure, le vecteur coût $g(a)$ est renseigné au milieu de chaque arc $a \in A$ et l'ensemble $H(n)$ est indiqué à côté de chaque nœud $n \in N$. Prenons $\delta = 0.1$ et $\varepsilon = \delta/(1-\delta)$. Supposons que les préférences du décideur sont représentables par la fonction $\mathrm{Ch}(u(x), v_0)$ *avec $v_0(\emptyset) = 0$, $v_0(\{1\}) = 0.6$, $v_0(\{2\}) = 0.8$, $v_0(\{1,2\}) = 1$ et $u_1(x) = u_2(x) = x/20$ pour tout $x \in \mathbb{R}_+$. Sans aucune information sur les préférences du décideur, l'ensemble Ω est composé de toutes les capacités v normalisées. Initialement, nous avons $O = \{\ell_s\}$ où $\ell_s = [s, \langle s \rangle, (0,0)]$. Durant la première itération, l'étiquette ℓ_s est déplacée de l'ensemble O vers l'ensemble C pour être développée. Comme nous avons $\Pi(s) = \{n_1, n_2, \gamma\}$, alors les trois étiquettes suivantes sont produites : $\ell_1 = [n_1, \langle s, n_1 \rangle, (7,9)]$, $\ell_2 = [n_2, \langle s, n_2 \rangle, (0,2)]$ et $\ell_\gamma = [\gamma, \langle s, \gamma \rangle, (10,12)]$. L'algorithme réalise alors le traitement suivant :*

- *Étiquette ℓ_γ : comme $S \cup O(\gamma) = \emptyset$, alors l'étiquette ℓ_γ est insérée dans O (cf. ligne 12) et n'est pas supprimée par la règle R3 (cf. ligne 13).*

- *Étiquette ℓ_1 : comme $O(n_1) \cup C(n_1) = \emptyset$, alors l'étiquette ℓ_1 est insérée dans O (cf. ligne 12). Cependant, comme $O(\gamma) \cup S = \{\ell_\gamma\}$, alors ℓ_1 est supprimée par la règle R3 car nous avons $\{g_{\ell_\gamma}\} \precsim^{\varepsilon}_{\Omega} f_1$ où $f_1 = g_{\ell_1} + (2,2) = (9,11)$ (cf. ligne 13).*
- *Étiquette ℓ_2 : comme $O(n_2) \cup C(n_2) = \emptyset$, alors l'étiquette ℓ_2 est ajoutée dans O (cf. ligne 12). De plus, cette étiquette n'est pas éliminée par la règle R3 car nous avons $\neg(\{g_{\ell_\gamma}\} \precsim^{\varepsilon}_{\Omega} f_2)$ où $f_2 = g_{\ell_2} + (4,12) = (4,14)$ (cf. ligne 13).*

Ainsi, $O = \{\ell_2, \ell_\gamma\}$ à la fin de la première itération. Notons que nous avons $F(\ell_2) = \{(4,14)\}$ et $F(\ell_\gamma) = \{(10,12)\}$. Comme aucun de ces vecteurs n'est Pareto-dominé, Choisir(O) peut retourner ℓ_2 ou ℓ_γ. Supposons que Choisir(O) retourne ℓ_γ à la deuxième itération. Dans ce cas, cette étiquette est retirée de l'ensemble O pour être insérée dans S (cf. ligne 7). Comme S contient un seul élément, alors $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) = 0 \leq (1+\varepsilon)\delta$. Par conséquent, S3 ne engendre ici aucune question. Durant la troisième itération, c'est l'étiquette ℓ_2 qui est développée par l'algorithme. Comme $\Pi(n_2) = \{n_1, \gamma\}$, alors ce développement engendre les étiquettes $\ell'_1 = [n_1, \langle s, n_2, n_1\rangle, (2,16)]$ et $\ell'_\gamma = [\gamma, \langle s, n_2, \gamma\rangle, (6,14)]$. L'algorithme réalise ensuite le traitement suivant :

- *Étiquette ℓ'_γ : comme $O(\gamma) = \emptyset$, $S = \{\ell_\gamma\}$ et $\neg(g_{\ell_\gamma} \precsim_P g_{\ell'_\gamma})$, alors l'étiquette ℓ'_γ est ajouté dans l'ensemble O. De plus, comme $\neg(\{g_{\ell_\gamma}\} \precsim^{\varepsilon}_{\Omega} f'_\gamma)$ où $f'_\gamma = g_{\ell'_\gamma} + (0,0) = (6,14)$, alors la règle R3 ne supprime pas cette étiquette (cf. ligne 13).*
- *Étiquette ℓ'_1 : rappelons que $O(n_1) = \emptyset$ et $C(n_1) = \{\ell_1\}$. De ce fait, comme $\neg(g_{\ell_1} \precsim_P g_{\ell'_1})$, alors ℓ'_1 est insérée dans l'ensemble O (cf. ligne 12). Par ailleurs, cette étiquette n'est pas éliminée par la règle R3 car nous avons $\neg(\{g_{\ell_\gamma}, g_{\ell'_\gamma}\} \precsim^{\varepsilon}_{\Omega} f'_1)$ où $f'_1 = g_{\ell'_1} + (2,4) = (4,18)$ (cf. ligne 13).*

Finalement, nous avons $O = \{\ell'_1, \ell'_\gamma\}$ à la fin de la troisième itération. Nous avons ici $F(\ell'_1) = \{(4,18)\}$ et $F(\ell'_\gamma) = \{(6,14)\}$. Comme aucun de ces vecteurs n'est Pareto-dominé, alors Choisir(O) peut retourner une de ces deux étiquettes à la quatrième itération. Supposons que Choisir(O) retourne l'étiquette ℓ'_γ. Dans ce cas, cette étiquette est retirée de l'ensemble O puis ajoutée dans l'ensemble S (cf. ligne 7). Comme $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) = \mathrm{mMR}_\varepsilon(\{(10,12),(6,14)\}, \Omega) > (1+\varepsilon)\delta$, alors la stratégie S3 demande au décideur de comparer les vecteurs coût $(10,12)$ et $(6,14)$ associés respectivement aux chemins $\langle s, \gamma\rangle$ et $\langle s, n_2, \gamma\rangle$. Comme $\mathrm{Ch}(u(10,12), v_0) = 0.58 \leq \mathrm{Ch}(u(6,4), v_0) = 0.62$, alors le décideur nous apprend qu'il préfère le premier chemin solution. Cette information induit la contrainte linéaire $\mathrm{Ch}(u(10,12), v) \leq \mathrm{Ch}(u(6,4), v)$ sur l'espace des paramètres admissibles, qui est équivalente à la contrainte $v(\{2\}) \geq 2/3$. Après cette mise à jour des capacités admissibles, nous obtenons $\mathrm{mMR}_\varepsilon(\{g_\ell : \ell \in S\}, \Omega) = \mathrm{MR}((10,12), \{g_\ell : \ell \in S\}, \Omega) \leq (1+\varepsilon)\delta$. Par conséquent, S3 ne pose plus aucune question et cette itération se termine. Comme $O = \{\ell'_1\}$, alors c'est l'étiquette ℓ'_1 qui est développée à la cinquième itération. Puisque nous avons $\Pi(n_1) = \{\gamma\}$, alors seule l'étiquette $\ell''_\gamma = [\gamma, \langle s, n_2, n_1, \gamma\rangle, (4,18)]$ est créée. Rappelons que $O(\gamma) = \emptyset$ et $S = \{\ell_\gamma, \ell'_\gamma\}$. De ce fait, comme $\neg(g_{\ell_\gamma} \precsim_P g_{\ell''_\gamma})$ et $\neg(g_{\ell'_\gamma} \precsim_P g_{\ell''_\gamma})$, alors ℓ'_γ est insérée dans l'ensemble O (cf. ligne 12). Toutefois, cette étiquette est immédiatement supprimée par la règle R3 car nous avons $\{g_{\ell_\gamma}, g_{\ell'_\gamma}\} \precsim^{\varepsilon}_{\Omega} f''_\gamma$ où $f''_\gamma = g_{\ell''_\gamma} + (0,0) = (4,18)$ (cf. ligne 13). Finalement, l'algorithme se termine parce que l'ensemble O est maintenant vide. Par construction, le chemin solution $\langle s, \gamma\rangle$ correspondant à l'étiquette $\ell_\gamma \in S$ est telle que $\mathrm{MR}(g_{\ell_\gamma}, \mathcal{X}, \Omega) \leq \delta$ à la fin de l'exécution.

v) Résultats expérimentaux

Dans cette sous-section, nous évaluons les performances de la méthode interactive que nous avons proposée pour l'utilité espérée à la Choquet, permettant de déterminer un chemin solution de vecteur coût $x \in \mathcal{X}$ vérifiant $\text{MR}(x, \mathcal{X}, \delta) \leq \delta$. Nous nous intéressons au temps d'exécution de cette méthode interactive [2] ainsi qu'au nombre de questions que celle-ci engendre. Cette méthode, nommée R^*_ε ci-après, consiste à combiner l'algorithme 5 avec la stratégie de génération de questions appelée S3. L'algorithme 5 est un algorithme approché raffinant l'algorithme MOA* avec la règle d'élagage R3, cette dernière étant fondée sur une relation de dominance à un facteur $(1+\varepsilon)$-près. La stratégie S3 est quant à elle construite à partir d'un concept de regret approché permettant de restaurer la garantie de performance à δ-près. Rappelons que la méthode interactive raffinant l'algorithme MOA* avec la règle d'élagage R2 (fondée sur la relation $\prec^\delta_\Omega$) et utilisant la stratégie S1 (avec le seuil δ) permet elle aussi de déterminer une solution de vecteur coût $x \in \mathcal{X}$ telle que $\text{MR}(x, \mathcal{X}, \delta) \leq \delta$. De ce fait, cette méthode combinant recherche exacte et élicitation sera considérée ici à titre de comparaison (celle-ci sera appelée R^* ci-après). Les résultats présentés dans cette sous-section ont tous été obtenus en moyennant les performances sur 30 tests.

Dans nos expériences, nous considérons des instances de graphe $G = (N, A)$ où les nœuds de $N \backslash \{s, \gamma\}$ sont engendrés uniformément sur la grille à deux dimensions $\{1, \ldots, 1000\} \times \{1, \ldots, 1000\}$. Le nœud source s et le nœud but γ sont quant à eux placés respectivement en position $(1, 500)$ et $(1000, 500)$. Chaque nœud est relié à 30 autres nœuds choisis aléatoirement, les vecteurs coût associés aux arcs étant engendrés à l'aide de fonctions Gaussiennes paramétrées par les distances Euclidiennes. Comme évaluation heuristique, nous considérons uniquement le point idéal $I(n) = (I_1(n), \ldots, I_q(n))$ pour tout $n \in N$, où $I_j(n) = \min_{x \in \{g(p) : p \in P(n,\gamma)\}} x_j$ pour tout $j \in \mathcal{Q}$. Rappelons que chaque valeur $I_j(n)$ peut être obtenue au préalable en appliquant l'algorithme A* au graphe évalué uniquement sur le critère j. Pour modéliser les préférences du décideur sur chaque critère, nous utilisons des fonctions de désutilité u_j, $j \in \mathcal{Q}$, de type sigmoïde (courbe en 'S') de la forme suivante : $u_j(x_j) = \frac{1}{1+\exp(-a_j(x_j - b_j))}$, où a_j et b_j sont des paramètres permettant de contrôler respectivement l'amplitude et la position du 'S' sur l'axe du critère j (cf. figure pour une illustration graphique).

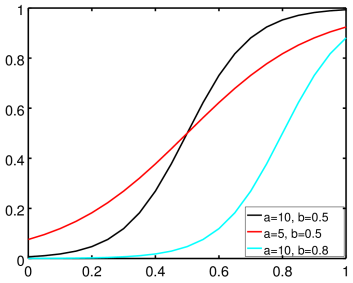


FIGURE 3.1 – Fonction $f(x) = \frac{1}{1+\exp(-a(x-b))}$ définie sur l'intervalle $[0, 1]$.

2. Les temps de calcul sont tous donnés en secondes, pour des expériences réalisées sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire. Les tests de dominance sont effectués par le solveur Gurobi depuis un programme écrit en Java.

Les réponses aux questions sont ici simulées en utilisant une utilité espérée à la Choquet $\mathrm{Ch}(u(x), v)$, où la capacité v est engendrée aléatoirement (rappelons que le système ne connaît pas la capacité utilisée pour répondre aux questions). Pour évaluer l'impact de la complexité du modèle, nous considérons deux sous-familles de ce modèle décisionnel. La première sous-famille correspond aux capacités additives (cf. définition 14), ce qui produit une modélisation avec un nombre de paramètres linéaire en le nombre de critères ; cette famille sera notée CA (pour Capacité Additive) ci-après. La seconde famille de fonctions correspond aux capacités 2-additives (cf. définition 19), permettant de modéliser des synergies entre deux critères quelconques avec un nombre quadratique de paramètres ; cette sous-famille sera notée C2A (pour Capacité 2-Additive) dans cette sous-section.

Les premiers tests effectués ont pour objectif de déterminer la valeur ε réalisant le meilleur compromis entre le temps de calcul et le nombre de questions. En effet, rappelons que toute valeur $\varepsilon \in [0, \delta/(1-\delta)]$ permet de garantir que notre méthode interactive retourne un vecteur solution x vérifiant $\mathrm{MR}(x, \mathcal{X}, \Omega) \leq \delta$ à la fin de l'exécution. Dans la table 3.3, nous faisons alors varier ε entre 0 et $\delta/(1-\delta)$, et nous voyons que le nombre de questions et les temps de calcul diminuent lorsque la valeur ε augmente. Ceci sous-entend que prendre $\varepsilon = \delta/(1-\delta)$ constitue la meilleure décision possible pour notre algorithme. De ce fait, nous utiliserons exclusivement cette valeur dans la suite de cette sous-section.

ε	0	$1/3 \times \delta/(1-\delta)$	$2/3 \times \delta/(1-\delta)$	$\delta/(1-\delta)$
temps	6.50	5.66	4.99	4.38
questions	13.80	12.53	11.63	10.43

TABLE 3.3 – Temps d'exécution de la méthode R^*_ε en secondes ($\delta = 0.2$, $q = 10$, $|N| = 1000$, C2A).

Dans la table suivante, nous comparons les performances des méthodes R^* et R^*_ε sur des petites instances, en faisant varier q le nombre de critères et δ le seuil de tolérance :

		$\delta = 0.01$		$\delta = 0.05$		$\delta = 0.1$	
q	méthode	temps	questions	temps	questions	temps	questions
2	R^*	0.6	3.0	7.9	1.9	309.7	1.0
2	R^*_ε	0.4	3.2	0.2	2.0	0.1	1.4
4	R^*	3.6	12.1	68.7	7.4	1537.0	4.4
4	R^*_ε	1.9	12.5	1.0	8.1	0.6	5.1

TABLE 3.4 – Comparaison des performances de R^* et R^*_ε ($|N| = 500$, C2A)

Dans la table 3.4, nous voyons que le temps d'exécution de R^* augmente drastiquement lorsque la valeur δ augmente. Plus précisément, en baissant nos exigences sur la qualité de la solution retournée, l'algorithme R^* engendre moins de questions mais s'exécute en temps moins raisonnables. Par exemple, pour $q = 4$, les temps de calcul sont multipliés par un facteur 400 lorsque que δ passe de 0.01 à 0.1. Ainsi, cette procédure ne semble réaliser aucun compromis entre le temps de résolution et le nombre

de questions posées. Observons que ce n'est pas du tout le cas pour notre méthode interactive. Cette différence s'explique par le fait que, en augmentant la valeur δ, nous réduisons à la fois le nombre d'étiquettes engendrées (car $\varepsilon = \delta/(1-\delta)$ croît avec δ) et le nombre de questions posées (car $\delta(1+\varepsilon)$ croît avec δ et ε). Par ailleurs, notons que la méthode R^*_ε est bien plus rapide que la méthode R^* ; par exemple, R^*_ε est 3000 fois plus rapide que R^* pour $q = 2$ et $\delta = 0.1$. De plus, l'écart entre ces deux méthodes se creuse en augmentant le nombre de critères et/ou la garantie de performance. De ce fait, la méthode R^* n'est pas envisageable pour résoudre de plus grandes instances.

Pour finir, nous étudions maintenant l'impact du nombre de paramètres du modèle décisionnel sur les performances de la méthode R^*_ε :

			$\delta = 0.05$		$\delta = 0.1$	
q	modèle	$\lvert N\rvert$	temps	questions	temps	questions
5	CA	500	1.1	8.1	0.7	5.6
5	C2A	500	2.9	13.5	1.5	8.9
5	CA	1000	1.4	8.3	0.9	5.6
5	C2A	1000	3.7	13.4	2.0	8.1
10	CA	500	7.8	22.3	3.4	13.7
10	C2A	500	66.9	47.5	18.1	27.2
10	CA	1000	10.0	26.9	4.3	16.0
10	C2A	1000	138.2	53.3	43.0	30.5

TABLE 3.5 – Performance de R^*_ε avec une capacité additive (CA) ou 2-additive (C2A).

Comme nous pouvions nous y attendre, le nombre de questions et le temps d'exécution sont plus importants pour la famille C2A que la famille CA. Ceci est le prix à payer pour obtenir de plus grandes possibilités descriptives et prescriptives. Toutefois, nous observons que les performances restent globalement correctes pour la famille C2A. Par exemple, pour l'instance la plus difficile ($q = 10$ et $|N| = 1000$), un vecteur solution x vérifiant $\mathrm{MR}(x, \mathcal{X}, \Omega) \leq \delta$ est déterminée en moins d'une minute avec 30 questions engendrées durant la résolution.

3.2 Problèmes d'arbres couvrants multicritères

Le problème d'arbres couvrants multicritères fait partie des problèmes classiques en optimisation combinatoire multicritère. Celui-ci apparaît naturellement dans des situations diverses, comme par exemple lorsque des entités quelconques doivent être reliées par un réseau de communication ou de transport. Dans ces situations, il est utile de pouvoir déterminer facilement une alternative de coût minimal, le coût étant généralement évalué selon différents aspects/critères comme le prix, le temps de construction ou encore la distance. Pour le cas particulier où un seul critère est considéré, il est bien connu que ce problème peut être résolu en temps polynomial par un algorithme glouton standard (e.g., [Kruskal, 1956, Prim, 1957]).

La généralisation classique de ce problème en décision multicritère concerne les graphes $G = (N, A)$ dont les arêtes sont évaluées par des vecteurs coût de $\mathbb{R}^q$, où q est le nombre de fonctions de coût (critères) à minimiser. Le problème d'arbres couvrants multicritères consiste à déterminer un arbre couvrant du graphe G par vecteur coût Pareto-optimal. Lorsque plusieurs critères sont pris en compte simultanément, le problème d'arbres couvrants devient intraitable (e.g., [Ehrgott and Gandibleux, 2000]) car il existe des instances de graphe $G = (N, A)$ pour lesquelles tous les arbres couvrants sont associés à un vecteur coût Pareto-optimal distinct des autres (e.g., [Hamacher and Ruhe, 1994]) ; comme il existe $|N|^{|N|-2}$ arbres couvrants dans un graphe complet possédant $|N|$ nœuds (e.g., [Cayley, 1889]), alors il est bien évidemment impossible de tous les lister en temps polynomial. Dans la littérature, il n'existe pas d'algorithme phare permettant de résoudre efficacement ce problème d'optimisation en pratique, sauf peut-être pour le cas bi-critère où des particularités peuvent être exploitées pour proposer des méthodes efficaces (e.g., [Sourd and Spanjaard, 2008, Stefan Ruzika, 2009]). Cette difficulté nous suggère d'utiliser des modèles décisionnels plus discriminants que la dominance de Pareto pour comparer les vecteurs coût solutions. Ainsi, nous supposons ici que les préférences du décideur sont représentables par une somme pondérée f_ω (cf. définition 11), où $\omega = (\omega_1, \ldots, \omega_q)$ est un vecteur de poids strictement positifs normalisés. Dans ce cadre, nous allons étudier le potentiel de l'élicitation incrémentale pour identifier la meilleure option pour le décideur.

3.2.1 Introduction et formalisation du problème

Considérons un graphe $G = (N, A)$ où chaque arête $a \in A$ est évaluée par un vecteur coût $x_a \in \mathbb{R}^q_+$ représentant le coût de cette arête sur l'ensemble des critères $\mathcal{Q} = \{1, \ldots, q\}$. Nous supposons ici que chaque critère admet une évaluation additive sur les arêtes : le coût du sous-ensemble d'arêtes $A' \subseteq A$ est défini par $x_{A'} = \sum_{a \in A'} x_a$. Un arbre couvrant T de G est un sous-graphe de G connexe et sans cycle qui contient tous les nœuds de G. Comme tout arbre couvrant T est complètement caractérisé par l'ensemble de ses arêtes, ces derniers pourront être confondus par abus de langage. L'ensemble des vecteurs coût associés aux arbres couvrants de G est noté $\mathcal{X}$ et représente l'image des solutions possibles dans l'espace des critères. Le problème d'arbres couvrants multicritères consiste à déterminer l'ensemble $\mathrm{ND}(\mathcal{X}) = \{x \in \mathcal{X} : \forall y \in \mathcal{X}, \neg(y \prec_P x)\}$ des vecteurs coût Pareto-optimaux et un arbre couvrant par vecteur de cet ensemble. Pour répondre à ce problème, il a été envisagé de généraliser des algorithmes de complexité polynomiale conçus pour le cas mono-critère. Malheureusement, nous allons voir que les généralisations multicritères proposées se sont confrontées à des difficultés inattendues et relativement ardues à surmonter :

Cas de l'algorithme de Prim

L'algorithme de Prim [Prim, 1957] est un algorithme glouton permettant de calculer en temps polynomial un arbre couvrant de coût minimal dans un graphe $G = (N, A)$ mono-évalué. Le principe général de cet algorithme est de faire croître un arbre depuis un sommet, en choisissant à chaque étape une arête de coût minimal parmi toutes celles qui sont adjacentes à l'arbre en construction (et ne formant

pas de cycle avec les arêtes de cet arbre). Cet algorithme se termine après $|N| - 1$ étapes car tout arbre couvrant possède exactement $|N| - 1$ arêtes. Lorsque les arêtes sont évaluées sur plusieurs critères, le choix de la prochaine arête à insérer dans l'arbre est moins évident, car l'ensemble des arêtes adjacentes Pareto-optimales est généralement composé de plusieurs éléments. Dans l'optique de déterminer $\mathrm{ND}(\mathcal{X})$ l'ensemble des vecteurs solutions Pareto-optimaux, il a été proposé de considérer, à chaque étape, toutes les prolongations possibles de l'arbre en construction par une arête Pareto-optimale adjacente ne créant pas de cycle (e.g., [Corley, 1985]). Plus précisément, au début de chaque étape i, nous avons maintenant un ensemble d'arbres T en construction composés de $i - 1$ arêtes chacun. À l'étape i, chaque arbre T est supprimé de l'ensemble pour laisser place à tous les arbres obtenus en ajoutant à T une arête Pareto-optimale parmi les arêtes adjacentes à T ne créant pas de cycle. En procédant ainsi, l'algorithme construit un ensemble d'arbres couvrants en $|N| - 1$ étapes, mais celui-ci est en réalité un sur-ensemble de $\mathrm{ND}(\mathcal{X})$, certains de ces arbres étant Pareto-dominés. Pour tenter de résoudre ce problème, il a été proposé de supprimer, au début de chaque étape i, tous les arbres en construction qui sont Pareto-dominés par d'autres arbres en construction (e.g., [Hamacher and Ruhe, 1994]). Bien que ce nouvel algorithme ait été repris dans d'autres travaux (e.g., [Zhou and Gen, 1999] et [Ehrgott, 2006] chapitre 7), celui-ci ne permet pas de répondre au problème rencontré de manière satisfaisante. En effet, il a été montré que cette nouvelle version de l'algorithme pouvait non seulement retourner des solutions Pareto-dominées mais aussi omettre des éléments de $\mathrm{ND}(\mathcal{X})$ (e.g., [Knowles and Corne, 2002]).

Cas de l'algorithme de Kruskal

L'algorithme de Kruskal [Kruskal, 1956] est un autre algorithme glouton permettant de déterminer un arbre couvrant de coût minimal dans un graphe $G = (N, A)$ mono-évalué. À partir d'un ensemble d'arêtes vide, celui-ci construit un arbre couvrant optimal en y ajoutant, à chaque étape, l'arête de coût minimal parmi toutes celles ne formant pas de cycle avec l'ensemble d'arêtes courant. En pratique, les arêtes du graphe sont tout d'abord triées par ordre croissant de coût, puis l'arbre est construit en sélectionnant les arêtes en suivant cet ordre de priorité. Quand le graphe est évalué sur plusieurs critères, il n'est généralement plus possible d'ordonner les arêtes de cette façon, car la dominance de Pareto peut laisser des arêtes incomparables. C'est pourquoi, il a été proposé de considérer tous les ordres compatibles avec la dominance de Pareto [Serafini, 1987], c'est-à-dire tous les ordres vérifiant la propriété suivante : pour toutes arêtes $a, a' \in A$, si $x_a \prec_P x_{a'}$ alors l'arête a est positionnée avant l'arête a'. En effet, il a été montré que, pour toute solution Pareto-optimale, il existe un ordre sur les arêtes compatible avec la dominance de Pareto pour lequel l'algorithme de Kruskal retourne cette solution [Serafini, 1987]. Cependant, en procédant ainsi, nous obtenons un sur-ensemble des vecteurs coût Pareto-optimaux car certains ordres peuvent conduire à des solutions Pareto-dominées.

Finalement, définir une généralisation multicritère d'algorithmes dédiés au cas mono-critère est une tâche relativement complexe pour ce problème de décision (en comparaison avec les problèmes de recherche dans un graphe d'états). Pour contourner ces difficultés, nous attaquons ici le problème d'arbres couvrants sous l'angle de la recherche fondée sur les préférences (e.g. [Perny and Spanjaard, 2005]). Plus

précisément, nous supposons que les préférences du décideur peuvent être représentées par une fonction d'agrégation $f_\omega : \mathbb{R}_+^q \Rightarrow \mathbb{R}_+$ de paramètres ω, mesurant le coût global de tout arbre couvrant. Dans ce cadre, il ne s'agit plus de déterminer l'ensemble $\mathrm{ND}(\mathcal{X})$ des vecteurs solutions Pareto-optimaux, mais de révéler un vecteur solution minimisant la fonction coût f_ω.

Dans cette section, nous supposons que la fonction f_ω est une somme pondérée (cf. définition 11), où $\omega = (\omega_1, \ldots, \omega_q)$ est un vecteur de poids strictement positifs normalisé. Si le jeu de poids ω est connu, alors le problème peut être résolu en temps polynomial. En effet, pour tout arbre couvrant T, nous avons :

$$f_\omega(x_T) = f_\omega(\sum_{a \in T} x_a) = \sum_{a \in T} f_\omega(x_a) \tag{3.2}$$

Ainsi, pour déterminer une solution f_ω-optimale, il suffit de remplacer le vecteur coût x_a par la valeur agrégée $f_\omega(x_a)$ pour toute arête $a \in A$, puis d'appliquer l'algorithme de Prim (ou Kruskal) sur le graphe mono-critère ainsi obtenu. Toutefois, il est relativement complexe de déterminer a priori le vecteur de poids ω permettant de représenter au mieux les préférences du décideur (en particulier lorsque les critères sont exprimés sur différentes échelles, comme le temps, la distance ou l'argent). C'est pourquoi, nous nous intéressons ici à des situations où le jeu de poids ω n'est pas connu de manière précise. À la place, nous considérons donc l'ensemble $\Omega = \{\omega \in \mathrm{int}(\mathbb{R}_+^q) : \sum_{j \in \mathcal{Q}} \omega_j = 1\}$ composé de tous les jeux de poids ω possibles, où int représente l'intérieur du cône. Chaque information de type "x est meilleur que y pour le décideur" impose la contrainte linéaire $f_\omega(x) \leq f_\omega(y)$ sur l'ensemble Ω des jeux de poids admissibles (Ω forme donc un polyèdre convexe).

Pour déterminer un arbre couvrant optimal, nous proposons ici une nouvelle approche consistant à combiner recherche et élicitation incrémentale des préférences. Le but est de limiter à la fois le nombre de questions nécessaires à la prise de décision et la taille de l'espace de recherche. Comme étape préliminaire, nous étudions la problématique de recherche des vecteurs solutions *potentiellement optimaux*, un vecteur solution étant potentiellement optimal si et seulement si celui-ci minimise la fonction f_ω pour au moins un jeu de poids admissible $\omega \in \Omega$. Puis, nous proposerons une méthode mêlant élicitation incrémentale et recherche, permettant de réduire progressivement l'ensemble Ω des jeux de poids admissibles durant la recherche, jusqu'à être en mesure de déterminer un arbre couvrant nécessairement optimal.

3.2.2 Calcul des solutions potentiellement optimales pour la somme pondérée

Dans cette sous-section, nous proposons une extension de l'algorithme de Prim [Prim, 1957] permettant de calculer l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ des vecteurs solutions potentiellement optimaux, formellement défini comme suit :

$$\mathrm{PO}_\Omega(\mathcal{X}) = \bigcup_{\omega \in \Omega} \arg\min_{x \in \mathcal{X}} f_\omega(x)$$

En d'autres termes, nous introduisons une méthode permettant de déterminer l'ensemble des vecteurs solutions $x \in \mathcal{X}$ qui minimisent la fonction f_ω pour au moins un jeu de poids $\omega \in \Omega$ admissible. Afin de décrire précisément cette méthode, nous introduisons la notion de *cocycle* utilisée en théorie des graphes :

Définition 52 (Cocycle)**.** *Le cocycle $C(N')$ d'un ensemble de nœuds $N' \subset N$ est défini par :*

$$C(N') = \{(n_1, n_2) \in A : n_1 \in N', n_2 \in N \backslash N'\}$$

Autrement dit, le cocycle $C(N')$ d'un ensemble de nœuds $N' \subset N$ est l'ensemble des arêtes ayant une extrémité dans l'ensemble N' et l'autre dans $N \backslash N'$. Avec cette terminologie, l'algorithme de Prim peut être décrit de la manière suivante :

- Initialisation : $T = \emptyset$ et $N' = \{n_0\}$, où n_0 est un nœud quelconque du graphe.
- Tant que $N' \neq N$, choisir une arête a de coût minimal dans le cocycle $C(N')$, ajouter cette arête à l'arbre T puis insérer dans N' l'extrémité de l'arête a qui n'appartient pas à l'ensemble N'.

L'algorithme que nous proposons pour la détermination de l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ est aussi un algorithme glouton commençant avec un ensemble $N' = \{n_0\}$ où n_0 est un nœud quelconque du graphe. La principale différence avec l'algorithme de Prim réside dans le fait que nous envisageons toutes les extensions possibles de l'arbre T avec une arête potentiellement optimale dans le cocycle $C(N')$. Par ailleurs, pour chaque arête $a \in C(N')$ potentiellement optimale, nous restreignons ensuite Ω à l'ensemble des jeux de poids admissibles ω tels que l'arête a minimise f_ω dans le cocycle $C(N')$. Cette procédure récursive, que nous nommons Ω-Prim, est présentée dans l'algorithme 6, l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ étant alors obtenu à partir de l'appel initial Ω-Prim$(\{n_0\}, \emptyset, \Omega)$. Dans cet algorithme, l'ensemble SOL contient à la fin un ensemble de paires (T, Ω_T) où T est arbre couvrant et Ω_T est un sous-ensemble des jeux de poids admissibles. Notons que la profondeur de l'arbre de recherche associé aux appels récursifs est exactement $|N| - 1$ car un nœud est ajouté à N' à chaque appel. Par ailleurs, rappelons que l'ensemble des arêtes potentiellement optimales dans le cocycle $C(N')$ peut être déterminé en temps polynomial en appliquant notre algorithme nommé Ω-Filter sur l'ensemble $\{x_a : a \in C(N')\}$ (cf. Section 3.1.2).

Algorithm 6: Ω-Prim(N', A', Ω')

Input: $N', A', \Omega', G = (N, A)$
if $N' = N$ **then**
  SOL $\leftarrow \{(A', \Omega')\}$
else
  SOL $\leftarrow \emptyset$
  Déterminer l'ensemble PO des arêtes potentiellement optimales dans $C(N')$
  for $a \in PO$ **do**
    $A'' \leftarrow A' \cup \{a\}$
    $N'' \leftarrow N' \cup \{n_a\}$, où n_a est l'extrémité de l'arête a telle que $n_a \notin N'$
    $\Omega'' \leftarrow \{\omega \in \Omega' : \forall a' \in \mathrm{PO},\ f_\omega(x_a) \leq f_\omega(x_{a'})\}$
    SOL $\leftarrow$ SOL $\cup$ Ω-Prim(N'', A'', Ω'')
  end
end
return SOL

Pour montrer que notre algorithme est valide, commençons par prouver que l'ensemble SOL retourné par l'appel initial Ω-Prim$(\{n_0\}, \emptyset, \Omega)$ contient au moins une paire (T, Ω_T) par élément de $\mathrm{PO}_\Omega(\mathcal{X})$:

Proposition 30. *Pour tous les vecteurs coût $x \in PO_\Omega(\mathcal{X})$, il existe une paire (T, Ω_T) dans l'ensemble SOL final tels que le vecteur coût x_T de l'arbre couvrant T vérifie $x_T = x$.*

Démonstration. Soit un arbre couvrant OPT associé à un vecteur coût potentiellement optimal. Montrons que notre algorithme retourne une paire (T, Ω_T) telle que $x_T = x_{OPT}$.

Par définition de $\mathrm{PO}_\Omega(\mathcal{X})$, nous savons qu'il existe $\omega_0 \in \Omega$ tel que $x_{OPT} \in \arg\min_{x \in \mathcal{X}} f_{\omega_0}(x)$. Notons G_{ω_0} le graphe obtenu en remplaçant le vecteur coût x_a par la valeur agrégée $f_{\omega_0}(x_a)$ pour toute arête $a \in A$. Comme $x_{OPT} \in \arg\min_{x \in \mathcal{X}} f_{\omega_0}(x)$, alors l'arbre OPT est un arbre couvrant de coût minimal dans le graphe mono-critère G_{ω_0} d'après l'équation (3.2). Par conséquent, il existe une exécution de l'algorithme de Prim sur le graphe mono-critère G_{ω_0} construisant l'arbre OPT à partir du nœud initial n_0. Notons $a^i, i \in \{1, \ldots, |N| - 1\}$, la $i^{\text{ème}}$ arrête sélectionnée durant cette exécution.

Durant le premier appel récursif de Ω-Prim, c'est le triplet $(N', A', \Omega') = (\{n_0\}, \emptyset, \Omega)$ qui est considéré. D'après la ligne 7, toute arête associée à un vecteur coût potentiellement optimal dans le cocycle $C(\{n_0\})$ est choisie pour étendre l'arbre A' (qui est vide ici). Puisque a^1 aurait été choisie par l'algorithme de Prim (appliqué sur le graphe G_{ω_0}) à cette étape, alors l'arête a^1 appartient au cocycle $C(\{n_0\})$ et l'inégalité $f_{\omega_0}(x_{a^1}) \leq f_{\omega_0}(x_a)$ est vraie pour tout $a \in C(\{n_0\})$. De ce fait, l'arête a^1 fait partie des arêtes potentiellement optimales dans le cocycle $C(\{n_0\})$. Ainsi, nous en déduisons que l'arête a^1 est sélectionnée par Ω-Prim pour étendre l'arbre A', ce qui nous donne le nouvel arbre $A'' = \emptyset \cup \{a^1\} = \{a^1\}$. D'après la ligne 10, l'algorithme Ω-Prim est ensuite appelé sur ce nouvel arbre. Pour ce nouvel appel récursif, l'ensemble Ω' est restreint aux jeux de poids ω tels que $f_\omega(x_{a^1}) \leq f_\omega(x_a)$ pour toute arête $a \in C(\{n_0\})$ (cf. ligne 9). Par définition, l'ensemble obtenu, noté Ω'', contient forcément le jeu de poids ω_0. Par conséquent, pour ce nouvel appel récursif, nous pouvons montrer similairement que l'arête a^2 sera choisie pour étendre l'arbre $\{a^1\}$. En itérant ce raisonnement, nous obtenons une séquence d'appels à Ω-Prim conduisant à l'arbre OPT, finalement retourné à la fin des appels récursifs (cf. ligne 2). □

Inversement, montrons que le vecteur coût x_T est un élément de $\mathrm{PO}_\Omega(\mathcal{X})$ pour toute paire (T, Ω_T) de l'ensemble SOL retourné par l'appel Ω-Prim$(\{n_0\}, \emptyset, \Omega)$.

Proposition 31. *Pour toute paire (T, Ω_T) de SOL final, l'arbre couvrant T est tel que $x_T \in PO_\Omega(\mathcal{X})$.*

Démonstration. Soit (T, Ω_T) une paire de l'ensemble SOL retourné suite à l'appel Ω-Prim$(\{n_0\}, \emptyset, \Omega)$. Nous voulons montrer que $x_T \in \mathrm{PO}_\Omega(\mathcal{X})$. Soit Ω-Prim(N_i, A_i, Ω_i), $i \in \{0, \ldots, |N| - 1\}$, la séquence d'appels ayant conduit à la paire (T, Ω_T). Rappelons que $(N_0, A_0, \Omega_0) = (\{n_0\}, \emptyset, \Omega)$. Par ailleurs, nous avons $(N_{|N|-1}, A_{|N|-1}, \Omega_{|N|-1}) = (N, T, \Omega_T)$ par définition.

Pour tout $i \in \{1, \ldots, |N| - 1\}$, notons a^i la $i^{\text{ème}}$ arête ajoutée à l'arbre T. D'après la ligne 7, nous avons $A_i = A_{i-1} \cup \{a^i\}$. Puis nous déduisons $\Omega_i = \{\omega \in \Omega_{i-1} : a^i \in \arg\min_{a \in C(N_{i-1})} f_\omega(x_a)\}$ avec la ligne 9. Par conséquent, nous avons $\Omega_{i+1} \subseteq \Omega_i$ pour tout $i \in \{0, \ldots, |N| - 2\}$, et en particulier $\Omega_{|N|-1} \subseteq \Omega_i$. Par ailleurs, comme Ω-Prim a été appelé sur le triplet (N_i, A_i, Ω_i), alors a^i est forcément

une arête potentiellement optimale dans le cocycle $C(N_{i-1})$ pour l'ensemble de jeux de poids Ω_{i-1}. Par conséquent, l'ensemble $\Omega_i = \{\omega \in \Omega_{i-1} : a^i \in \arg\min_{a \in C(N_{i-1})} f_\omega(x_a)\}$ n'est pas vide, et en particulier $\Omega_{|N|-1} \neq \emptyset$. Soit $\omega_0 \in \Omega_{|N|-1}$. Comme $\Omega_{|N|-1} \subseteq \Omega_i$ pour tout $i \in \{1, \ldots, |N|-1\}$, alors l'arête a^i minimise forcément la fonction f_{ω_0} sur le cocycle $C(N_{i-1})$. Par conséquent, il existe une exécution de Prim sur le graphe mono-critère G_{ω_0} construisant l'arbre T à partir du nœud n_0, où G_{ω_0} est le graphe obtenu en remplaçant x_a par $f_{\omega_0}(x_a)$ pour toute arête $a \in A$. De ce fait, nous pouvons déduire que $x_T \in \arg\min_{x \in \mathcal{X}} f_{\omega_0}(x)$. Comme $\omega_0 \in \Omega$, alors nous en concluons que $x_T \in \mathrm{PO}_\Omega(\mathcal{X})$. □

Les deux propositions précédentes permettent de garantir que l'ensemble des vecteurs coût retourné par l'appel Ω-Prim$(\{n_0\}, \emptyset, \Omega)$ correspond exactement à l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$. Par ailleurs, la première composante de chaque paire (T, Ω_T) retournée fournit un arbre couvrant pour chaque vecteur coût potentiellement optimal. Les ensembles Ω_T contenus dans ces paires permettent quant à eux de définir les jeux de poids ω pour lesquels l'arbre T est optimal ; notons que leur union couvre tout l'ensemble Ω.

Cependant, lorsque l'ensemble Ω des jeux de poids possibles est très grand, l'ensemble $\mathrm{PO}_\Omega(\mathcal{X})$ des vecteurs solutions potentiellement optimaux peut contenir un nombre d'éléments trop élevé. En particulier, lorsque l'ensemble Ω est composé de tous les jeux de poids possibles, il existe des graphes complets pour lesquels tous les arbres couvrants réalisables sont associés à des vecteurs coûts potentiellement optimaux distincts (e.g., [Hamacher and Ruhe, 1994]). Pour de tels graphes, l'ensemble retourné par Ω-Prim est donc de taille exponentielle en le nombre de nœuds du graphe. C'est pourquoi nous nous intéressons maintenant à la détermination d'une solution préférée par élicitation incrémentale des préférences.

3.2.3 Un algorithme glouton interactif pour la détermination d'une solution optimale avec une somme pondérée

Dans les situations où le décideur peut répondre à nos questions, nous pouvons envisager de l'interroger pour réduire l'ensemble Ω de manière à identifier un vecteur solution *nécessairement optimal* : le vecteur $x \in \mathcal{X}$ est dit nécessairement optimal si et seulement si $f_\omega(x) \leq f_\omega(y)$ pour tout vecteur solution $y \in \mathcal{X}$ et tout jeu de poids $\omega \in \Omega$. Lorsque l'ensemble SOL retourné par l'appel Ω-Prim$(\{n_0\}, \emptyset, \Omega)$ contient une paire (T, Ω_T) telle que $f_\omega(x_T) \leq f_\omega(x_{T'})$ pour tout $\omega \in \Omega$ et toute paire $(T', \Omega_T) \in \mathrm{SOL}$, alors le vecteur solution x_T est en réalité nécessairement optimal. En effet, par définition de $\mathrm{PO}_\Omega(\mathcal{X})$, nous en déduisons que l'arbre T vérifie $f_\omega(x_T) \leq f_\omega(x)$ pour tout $\omega \in \Omega$ et tout $x \in \mathcal{X}$. Ainsi, si l'ensemble SOL contient une paire (T, Ω_T) telle que $\mathrm{MR}(x_T, \{x_{T'} : (T', \Omega_{T'}) \in \mathrm{SOL}\}, \Omega) \leq 0$, alors nous savons que le vecteur x_T est nécessairement optimal sans avoir à poser des questions au décideur. Dans le cas contraire, nous pouvons être tentés de l'interroger à la fin de la recherche tant que la valeur $\mathrm{MR}(x_T, \{x_{T'} : (T', \Omega_{T'}) \in \mathrm{SOL}\}, \Omega) > 0$. Cependant, une telle approche ne serait pas très efficace en pratique car l'ensemble des vecteurs solutions potentiellement optimaux peut être de grande taille. À la place, nous proposons une nouvelle fois de combiner recherche et élicitation incrémentale pour limiter le nombre d'arbres couvrants construits ainsi que le nombre de questions engendrées.

Nous proposons maintenant un algorithme glouton interactif permettant de construire un arbre couvrant T tel que $\mathrm{MR}(x_T, \mathcal{X}, \Omega) \leq \delta$, où $\delta \geq 0$ est un seuil de tolérance que le décideur considère acceptable (si $\delta = 0$, alors x est nécessairement optimal). Afin de limiter les interventions du décideur, il est nécessaire de choisir soigneusement quand lui poser des questions durant la résolution. Notre proposition consiste à collecter des informations sur les préférences du décideur uniquement pour discriminer entre les arêtes du cocycle $C(N')$. Plus précisément, pour choisir la $i^{\text{ème}}$ arête à insérer dans l'arbre, nous lui posons des questions jusqu'à ce que la quantité $\mathrm{mMR}(\{x_a : a \in C(N')\}, \Omega)$ devienne inférieure à la valeur δ_i représentant une fraction du seuil de tolérance δ (i.e. $\sum_{i=1}^{|N|-1} \delta_i = \delta$). À ce moment là, l'arbre est étendu avec une arête a du cocycle qui minimise la valeur $\mathrm{MR}(x_a, \{x_{a'} : a' \in C(N')\}, \Omega)$. Notre méthode de recherche interactive est détaillée dans l'algorithme 7. Pour déterminer la prochaine question à poser, nous pouvons par exemple suivre la stratégie de questions CSS présentée en Section 1.4.2. Dans ce cas, le nombre d'itérations de la boucle "while" est borné supérieurement par la quantité $|C(N')| \leq |N|^2$, ce qui assure que notre algorithme est de complexité polynomiale et pose un nombre polynomial de questions dans le pire cas.

Algorithm 7:

Input: $G = (N, A)$; $n_0 \in N$; $\delta_i, i \in \{1, \ldots, |N| - 1\}$: valeurs positives sommant à δ

1 $N' \leftarrow \{n_0\}$
2 $A' \leftarrow \emptyset$
3 **for** $i = 1 \ldots |N| - 1$ **do**
4 $\quad \mathrm{ND} \leftarrow \{a \in C(N') : \forall a' \in C(N'), \mathrm{not}(x_{a'} \prec_P x_a)\}$
5 $\quad X \leftarrow \{x_a : a \in \mathrm{ND}\}$
6 $\quad$ **while** $\mathrm{mMR}(X, \Omega) > \delta_i$ **do**
7 $\quad\quad$ Poser une question au décideur
8 $\quad\quad$ Mettre à jour l'ensemble Ω en ajoutant la contrainte linéaire associée à sa réponse
9 $\quad$ **end**
10 $\quad$ Choisir une arête $a \in \mathrm{ND}$ telle que $x_a \in \arg\min_{x \in X} \mathrm{MR}(x, X, \Omega)$
11 $\quad A' \leftarrow A' \cup \{a\}$
12 $\quad N' \leftarrow N' \cup \{n_a\}$, où n_a est l'extrémité de l'arête a telle que $n_a \notin N'$
13 **end**
14 **return** A'

La validité de notre algorithme est établie par la proposition suivante :

Proposition 32. *L'arbre couvrant T retourné par l'algorithme 7 vérifie $\mathrm{MR}(x_T, \mathcal{X}, \Omega_f) \leq \delta$, où Ω_f est l'ensemble des jeux de poids admissibles à la fin de l'exécution.*

Démonstration. Montrer que l'inégalité $\mathrm{MR}(x_T, \mathcal{X}, \Omega_f) \leq \delta$ est vraie revient à prouver que l'inégalité $\mathrm{PMR}(x_T, x_{T_0}, \Omega_f) \leq \delta$ est satisfaite pour tout arbre couvrant T_0 du graphe G. Pour chaque itération $i \in \{1, \ldots, |N| - 1\}$, notons a^i l'arête ajoutée à l'arbre construit (cf. ligne 11), N^i l'ensemble de nœuds N' (cf. ligne 12) et Ω_i l'ensemble des jeux de poids admissibles à la fin de l'itération i.

Soit $j \in \{1, \ldots, |N|-1\}$ la première itération telle que $a^j \notin T_0$. Puisque T_0 est un arbre couvrant de G, nous savons qu'il existe une chaîne c dans T_0 reliant les deux extrémités de a^j. Par ailleurs, comme l'arête $a^j \in C(N^{j-1})$, alors a^j relie un nœud de N^{j-1} à un nœud de $N \backslash N^{j-1}$. Par conséquent, il existe une arête b^j dans la chaîne c qui relie un nœud de N^{j-1} à un nœud de $N \backslash N^{j-1}$. Ainsi, il existe une arête b^j dans la chaîne c qui appartient au cocycle $C(N^{j-1})$. Deux cas sont alors à distinguer : soit $b^j \in \mathrm{ND}_j$ soit $b^j \notin \mathrm{ND}_j$, où ND_j représente l'ensemble des arêtes non Pareto-dominées dans le cocycle $C(N^{j-1})$ (cf. ligne 4). Montrons que, dans les deux cas, nous avons $\mathrm{PMR}(x_{a^j}, x_{b^j}, \Omega_f) \leq \delta_j$:

- *Cas où $b^j \in ND_j$:* comme l'arête a^j a été sélectionnée pour construire l'arbre T à l'itération j, alors nous avons $x_{a^j} \in \arg\min_{x \in X_j} \mathrm{MR}(x, X_j, \Omega_j)$ où $X_j = \{x_a : a \in \mathrm{ND}_j\}$ (cf. ligne 10). Par ailleurs, comme nous avons $\mathrm{mMR}(X_j, \Omega_j) \leq \delta_j$ à la fin de la boucle "while" (cf. ligne 6), alors nous avons nécessairement $\mathrm{PMR}(x_{a^j}, x_{b^j}, \Omega_j) \leq \delta_j$. Enfin, puisque $\Omega_j \subseteq \Omega_f$ par construction (cf. ligne 8), alors nous en concluons $\mathrm{PMR}(x_{a^j}, x_{b^j}, \Omega_f) \leq \delta_j$.
- *Cas où $b^j \notin ND_j$:* par définition de ND_j, il existe une arête $a \in \mathrm{ND}_j$ telle que $x_a \prec_P x_{b^j}$. Puisque la somme pondérée est un opérateur monotone, alors nous avons forcément $f_\omega(x_a) \leq f_\omega(x_{b^j})$ pour tout jeu de poids $\omega \in \Omega_f$. Par conséquent, nous avons $\mathrm{PMR}(x_{a^j}, x_a, \Omega_f) \geq \mathrm{PMR}(x_{a^j}, x_{b^j}, \Omega_f)$. Par ailleurs, comme $x_a \in \mathrm{ND}_j$, alors nous avons $\mathrm{PMR}(x_{a^j}, x_a, \Omega_f) \leq \delta_j$ (cf. cas précédent). Par transitivité, nous obtenons $\mathrm{PMR}(x_{a^j}, x_{b^j}, \Omega_f) \leq \delta_j$.

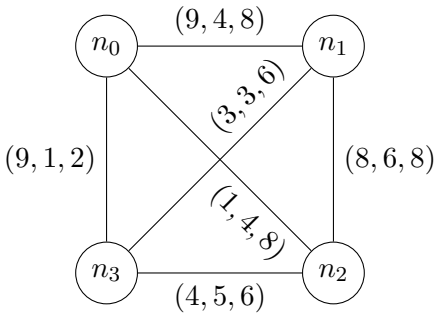
À présent, considérons l'arbre couvrant T_1 obtenu en remplaçant l'arête b^j de T_0 par l'arrête a^j. Soit $k \in \{j+1, \ldots, |N|-1\}$ la première itération telle que $a^k \notin T_1$. De manière similaire au cas de T_0, nous pouvons montrer que $\mathrm{PMR}(x_{a^k}, x_{b^k}, \Omega_f) \leq \delta_k$. Plus précisément, en itérant ce raisonnement, nous obtenons $\mathrm{PMR}(x_{a^l}, x_{b^l}, \Omega_f) \leq \delta_l$ pour toute itération $l \in \{1, \ldots, |N|-1\}$ telle que $a^l \notin T_0$. Par ailleurs, pour toute itération $l \in \{1, \ldots, |N|-1\}$ telle que $a^l \in T_0$, nous avons $\mathrm{PMR}(x_{a^l}, x_{a^l}, \Omega_f) = 0 \leq \delta_l$. Considérons alors la bijection $\pi : T \to T_0$ telle que $\pi(a^l) = b^l$ si $a^l \notin T_0$ et $\pi(a^l) = a^l$ sinon. Par construction, nous avons donc $\mathrm{PMR}(x_{a^l}, x_{\pi(a^l)}, \Omega_f) \leq \delta_l$ pour tout $l \in \{1, \ldots, |N|-1\}$. Ainsi :

$$
\begin{aligned}
\mathrm{PMR}(x_T, x_{T_0}, \Omega_f) &= \max_{\omega \in \Omega_f} \Big\{ f_\omega(x_T) - f_\omega(x_{T_0}) \Big\} \\
&= \max_{\omega \in \Omega_f} \Big\{ f_\omega(\sum_{l=1}^{|N|-1} x_{e_l}) - f_\omega(\sum_{l=1}^{|N|-1} x_{\pi(e_l)}) \Big\} \\
&= \max_{\omega \in \Omega_f} \sum_{l=1}^{|N|-1} \Big(f_\omega(x_{e_l}) - f_\omega(x_{\pi(e_l)}) \Big) \text{ car } f_\omega \text{ est linéaire} \\
&\leq \sum_{l=1}^{|N|-1} \max_{\omega \in \Omega_f} \Big\{ f_\omega(x_{e_l}) - f_\omega(x_{\pi(e_l)}) \Big\} \\
&= \sum_{l=1}^{|N|-1} \mathrm{PMR}(x_{e_l}, x_{\pi(e_l)}, \Omega_f) \\
&\leq \sum_{l=1}^{|N|-1} \delta_l \\
&= \delta
\end{aligned}
$$

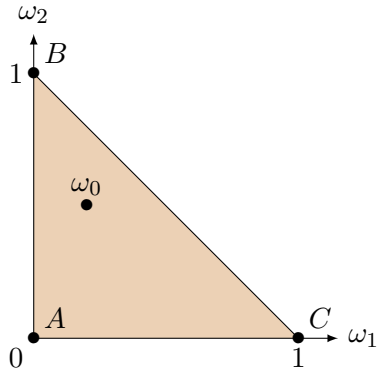
Finalement, nous avons montré que $\mathrm{PMR}(x_T, x_{T_0}, \Omega_f) \leq \delta$, ce qui permet de conclure la preuve. □

L'exemple suivant présente une exécution de l'algorithme 7 combiné avec la stratégie de questions CSS sur une instance de petite taille :

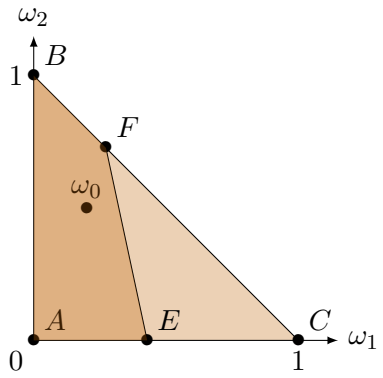
Exemple 26. *Considérons le graphe complet $G = (N, A)$ suivant :*



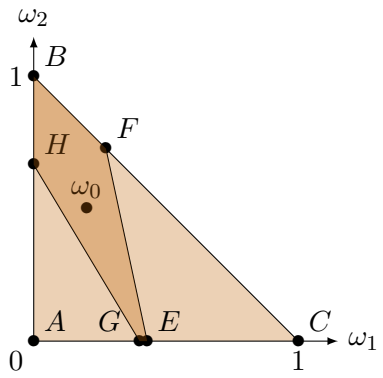
Sur ce graphe, le coût $x_a \in \mathbb{R}^3_+$ de l'arête a est indiqué à côté de chaque arête $a \in A$. Supposons que les préférences du décideur soient représentables par une somme pondérée f_{ω_0} avec $\omega_0 = (0.2, 0.5, 0.3)$. Sans aucune information sur les préférences du décideur, l'ensemble Ω des jeux de poids $\omega = (\omega_1, \omega_2, \omega_3)$ admissibles correspond à l'intérieur du triangle ABC représenté dans l'espace (ω_1, ω_2) ci-dessous, le poids ω_3 étant défini implicitement par la contrainte de normalisation (i.e. $\omega_3 = 1 - \omega_1 - \omega_2$) :



Déroulons l'algorithme 7 sur cet exemple avec un seuil de tolérance $\delta = 0$ et la stratégie de questions CSS (présentée en Section 1.4.2). Au départ, nous avons $N' = \{n_0\}$ et $A' = \emptyset$. Durant l'itération $i = 1$, le cocycle $C(N')$ est constitué des arêtes (n_0, n_1), (n_0, n_2) et (n_0, n_3). Les arêtes non Pareto-dominées sont (n_0, n_2) et (n_0, n_3) de vecteurs coûts $(1, 4, 8)$ et $(9, 1, 2)$ respectivement. Nous avons ici $\mathrm{MR}((1, 4, 8), \{(1, 4, 8), (9, 1, 2)\}, \Omega) \approx 6$ *et* $\mathrm{MR}((9, 1, 2), \{(1, 4, 8), (9, 1, 2)\}, \Omega) \approx 8$. *Par conséquent, nous avons* $\mathrm{mMR}(\{(1, 4, 8), (9, 1, 2)\}, \Omega) \approx 6 > \delta_1 = 0$. *Par conséquent, la condition de la boucle "while" est vérifiée. La stratégie CSS nous dicte alors de demander au décideur de comparer les vecteurs coûts $(1, 4, 8)$ et $(9, 1, 2)$. Le décideur nous répond ici que $(9, 1, 2)$ est meilleur que $(1, 4, 8)$ car $f_{\omega_0}(9, 1, 2) = 2.9 \leq f_{\omega_0}(1, 4, 8) = 4.6$. L'algorithme met alors à jour l'ensemble Ω des jeux de poids admissibles en insérant la contrainte linéaire $f_\omega(9, 1, 2) \leq f_\omega(1, 4, 8)$ représentée par la droite (EF) sur la figure suivante, la partie admissible étant à gauche de cette droite :*



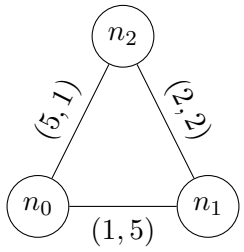
À présent, Ω *correspond à l'intérieur du polygone* $ABFE$ *auquel on ajoute le segment ouvert* $]EF[$*. De ce fait, nous obtenons* $\mathrm{MR}((1,4,8),\{(1,4,8),(9,1,2)\},\Omega) \approx 6$ *et* $\mathrm{MR}((9,1,2),\{(1,4,8),(9,1,2)\},\Omega) = 0$*. Nous avons donc maintenant* $\mathrm{mMR}(\{(1,4,8),(9,1,2)\},\Omega) = 0 \leq \delta_1 = 0$ *et donc la boucle "while" se termine. L'arrête* (n_0,n_3) *correspondant au vecteur coût* $(9,1,2)$ *est ensuite insérée dans l'arbre* A' *et le sommet* n_3 *est ajouté à l'ensemble* N'*. À l'itération* $i = 2$*, le cocycle* $C(N')$ *est constitué des arêtes* (n_0,n_1)*,* (n_0,n_2)*,* (n_3,n_1) *et* (n_3,n_2)*. Les arêtes non Pareto-dominées sont* (n_0,n_2) *et* (n_3,n_1) *de vecteurs coûts* $(1,4,8)$ *et* $(3,3,6)$ *respectivement. Nous avons ici* $\mathrm{MR}((1,4,8),\{(3,3,6),(1,4,8)\},\Omega) \approx 2$ *et* $\mathrm{MR}((3,3,6),\{(3,3,6),(1,4,8)\},\Omega) = 0$*. Par conséquent, nous avons* $\mathrm{mMR}(\{(3,3,6),(1,4,8)\},\Omega) = 0 \leq \delta_2 = 0$*. Ainsi, la condition de la boucle "while" n'est pas vérifiée ici et donc aucune question n'est posée à cette itération. L'arête* (n_3,n_1) *correspondant au vecteur coût* $(3,3,6)$ *est ajoutée à l'arbre* A' *et le sommet* n_1 *est inséré dans l'ensemble* N'*. À l'itération* $i = 3$*, le cocycle* $C(N')$ *est constitué des arêtes* (n_0,n_2)*,* (n_1,n_2) *et* (n_3,n_2)*. Les arêtes non Pareto-dominées sont* (n_0,n_2) *et* (n_3,n_2) *de vecteurs coûts* $(1,4,8)$ *et* $(4,5,6)$ *respectivement. Nous avons ici les regrets suivants :* $\mathrm{MR}((1,4,8),\{(4,5,6),(1,4,8)\},\Omega) \approx 2$ *et* $\mathrm{MR}((4,5,6),\{(4,5,6),(1,4,8)\},\Omega) \approx 1.55$*. Ainsi, nous avons* $\mathrm{mMR}(\{(4,5,6),(1,4,8)\},\Omega) \approx 1.55 > \delta_3 = 0$*. Par conséquent, la condition de la boucle "while" est vérifiée et donc nous devons poser au moins une question au décideur. La stratégie CSS nous suggère de demander au décideur de comparer les vecteurs* $(1,4,8)$ *et* $(4,5,6)$*. Celui-ci nous répond alors que* $(1,4,8)$ *est meilleur que* $(4,5,6)$ *car nous avons* $f_{\omega_0}(1,4,8) = 4.6 \leq f_{\omega_0}(4,5,6) = 5.1$*. L'algorithme met alors à jour l'ensemble des jeux de poids admissibles* Ω *en ajoutant la contrainte linéaire* $f_\omega(1,4,8) \leq f_{\omega_0}(4,5,6)$ *qui est représentée par la droite* (GH) *sur la figure suivante, la partie admissible étant la zone située à droite de* (GH) *:*



L'ensemble Ω *correspond maintenant à l'intérieur du polygone* $BEFGH$ *auquel il faut ajouter les segments ouverts* $]EF[$ *et* $]GH[$. *Après cette mise à jour, nous obtenons* $\mathrm{MR}((1,4,8),\{(4,5,6),(1,4,8)\},\Omega) = 0$ *et* $\mathrm{MR}((4,5,6),\{(4,5,6),(1,4,8)\},\Omega) \approx 1.55$. *Nous avons donc* $\mathrm{mMR}(\{(4,5,6),(1,4,8)\},\Omega) = 0 \leq \delta_3 = 0$. *L'arête* (n_0,n_2) *est alors ajoutée à l'arbre* A' *et le sommet* n_2 *est inséré dans l'ensemble* N'. *Finalement, l'algorithme s'arrête en retournant l'arbre* $A' = \{(n_0,n_3),(n_3,n_1),(n_0,n_2)\}$. *Ainsi, nous sommes parvenus à détecter le meilleur arbre couvrant pour le décideur avec seulement deux questions.*

Remarquons que, dans l'exemple précédent, l'arbre couvrant retourné correspond nécessairement à un vecteur coût situé sur les bords de l'enveloppe convexe des vecteurs solutions puisque cet arbre est optimal au sens d'une somme pondérée f_ω (avec $\omega = (0.2, 0.5, 0.3)$ par exemple). Ceci est toujours vrai avec $\delta = 0$. Néanmoins, lorsque $\delta > 0$, notre algorithme peut produire une solution Pareto-optimale à l'intérieur de l'enveloppe convexe des solutions réalisables. Ceci s'explique par l'utilisation du critère de décision Minimax Regret, comme le montre l'exemple simple suivant :

Exemple 27. *Soit* $G = (N, A)$ *le graphe complet suivant :*



Ce graphe admet uniquement les trois arbres couvrants suivants : $\{(n_0,n_1),(n_1,n_2)\}$, $\{(n_0,n_2),(n_1,n_2)\}$ *et* $\{(n_0,n_1),(n_0,n_2)\}$. *Les vecteurs solutions correspondant sont respectivement* $(3,7)$, $(7,3)$ *et* $(6,6)$. *Il est facile de vérifier qu'il n'existe aucun jeu de poids* ω *tel que* $f_\omega(6,6) \leq f_\omega(3,7)$ *et* $f_\omega(6,6) \leq f_\omega(7,3)$. *Pourtant, lorsque* Ω *contient tous les jeux de poids possibles, nous obtenons les regrets suivants :* $\mathrm{MR}((6,6),\Omega) \approx 3$, $\mathrm{MR}((3,7),\Omega) \approx 4$ *et* $\mathrm{MR}((7,3),\Omega) \approx 4$. *Par conséquent, le vecteur coût* $(6,6)$ *est associé à l'unique arbre couvrant optimal au sens du critère de décision Minimax Regret.*

L'exemple ci-dessus nous montre que l'utilisation d'une somme pondérée avec un jeu de poids imprécis peut conduire à la recommandation d'une solution à l'intérieur de l'enveloppe des solutions réalisables, ce qui améliore les capacités prescriptives de ce modèle décisionnel.

3.2.4 Résultats expérimentaux

Dans cette sous-section, nous évaluons les performances des algorithmes 6 et 7 sur des graphes multicritères engendrés aléatoirement. Les tests ont été réalisés sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire et les optimisations linéaires ont été effectuées par le solveur Gurobi depuis un programme écrit en Java. Pour l'algorithme 6 permettant de déterminer toutes les solutions potentiellement optimales, nous considérons des graphes à 10 nœuds dont la densité varie entre 50% et 100%. Rappelons qu'un graphe complet (i.e. de densité 100%) avec 10 nœuds admet exactement 10^8 arbres couvrants. Dans ces

expériences, le nombre de critères q varie de 2 à 5 et les coûts sont tirés aléatoirement dans $\{1, \ldots, 1000\}^q$. Afin d'évaluer l'impact de l'ensemble Ω, nous faisons aussi varier p le nombre de données de type "je préfère x à y" disponibles avant l'exécution de cet algorithme (données engendrées aléatoirement). Les moyennes des temps de calcul observés sont données dans la table 3.6.

q	densité	$p = 0$	$p = 1$	$p = 2$
2	50%	0.15	0.12	0.09
3	50%	0.89	0.78	0.75
4	50%	10.70	9.82	8.03
5	50%	17.79	17.40	15.73
2	75%	0.42	0.36	0.27
3	75%	8.60	5.41	3.16
4	75%	17.40	16.23	14.41
5	75%	267.93	219.25	168.83
2	100%	0.54	0.46	0.28
3	100%	12.95	10.35	9.34
4	100%	91.48	75.45	69.55
5	100%	668.21	634.26	621.46

TABLE 3.6 – Temps de calcul moyens de l'algorithme 6 donnés en secondes ($|N| = 10$, 30 tests).

Dans cette table, nous voyons que les temps de calculs diminuent progressivement avec la taille de l'ensemble Ω mais augmentent drastiquement avec le nombre de critères. Néanmoins, sur toutes les instances considérées, notre algorithme est capable de déterminer toutes les solutions potentiellement optimales en moins de 12 minutes. Ainsi, notre algorithme est capable de déterminer relativement rapidement toutes les solutions situées sur les bords de l'enveloppe convexe des solutions réalisables. Rappelons que la détermination de ces solutions est souvent considérée comme étant une base utile pour une exploration plus approfondie de l'ensemble des solutions Pareto-optimales dans les méthodes en deux phases standards en optimisation multicritère (e.g., Ehrgott [2006]).

Dans les expériences précédentes, nous avons constaté que les données de préférences permettent de réduire assez timidement les temps de résolution quand ces dernières sont tirées aléatoirement avant la résolution (cf. table 3.6). Cependant, nous allons voir maintenant que combiner résolution et élicitation incrémentale permet de réduire drastiquement les temps de calcul, nous permettant de traiter de plus grandes instances. Plus précisément, pour l'algorithme 7 combiné à la stratégie CSS, nous considérons des graphes contenant entre 25 et 100 nœuds (densité 50%) et un nombre de critères q variant de 2 à 8. Afin d'estimer l'impact du seuil de tolérance δ sur les temps de calcul et le nombre de questions posées, nous reportons dans la table 3.7 les résultats obtenus pour les valeurs suivantes : $\delta = 0.05$ et $\delta = 0.1$; les valeurs δ_i sont ici définies par $\delta_i = \delta/(|N| - 1)$. Dans ces expériences, les réponses aux questions ont été simulées par le biais d'une somme pondérée f_ω dont le jeu de poids ω a été engendré aléatoirement avant de commencer la résolution.

		$\delta = 0.05$		$\delta = 0.1$	
n	$\|V\|$	temps	questions	temps	questions
2	25	0.9	3.8	1.1	3.4
2	50	2.0	3.6	2.9	2.8
2	75	5.4	4.4	7.0	3.4
2	100	8.3	4.2	10.7	2.8
4	25	4.6	16.2	7.2	11.8
4	50	23.5	17.0	42.6	12.6
4	75	75.8	19.8	116.5	12.2
4	100	127.1	20.4	227.7	12.1
6	25	12.9	29.2	13.6	15.4
6	50	81.1	32.0	123.2	22.8
6	75	256.8	30.8	498.1	25.0
6	100	750.3	33.0	1182.4	25.4
8	25	32.2	49.2	40.3	34.6
8	50	234.6	62.0	370.8	35.2
8	75	731.0	65.6	1105.8	35.8
8	100	2518.9	71.4	3389.0	40.8

TABLE 3.7 – Performances de l'algorithme 7 avec la CSS (temps moyens donnés en secondes, 30 tests).

Dans cette table, nous observons que notre procédure interactive est très efficace au vu de la nature combinatoire de l'espace des solutions et du nombre de critères considérés. En comparaison, la littérature sur le calcul des arbres couvrants Pareto-optimaux ne considère pas plus de deux ou trois critères. L'efficacité relative de notre procédure s'explique par la possibilité de collecter des informations sur les préférences du décideur pendant la résolution. Ceci permet en effet de déterminer rapidement une alternative Pareto-optimale que le décideur juge pertinente. À titre d'exemple, notre procédure ne requiert pas plus de 12 questions en moyenne pour déterminer un arbre couvrant presque optimal aux yeux du décideur (avec $\delta = 0.1$) pour des instances à 100 nœuds et 4 critères. Nous observons aussi que le nombre de questions réduit drastiquement lorsque la garantie de performance est affaiblie (de 0.5 à 0.1), mais que cela provoque une augmentation des temps de calculs car les arêtes potentiellement optimales dans les cocycles sont plus nombreuses.

3.3 Conclusion du chapitre

Dans ce chapitre, nous avons étudié le potentiel de la résolution par élicitation incrémentale dans le cadre de deux problèmes d'optimisation combinatoire multi-objectifs : la recherche dans un graphe d'états multicritère et le problème d'arbres couvrants multicritères. Les méthodes proposées dans ce chapitre supposent que les préférences du décideur sont représentables par une fonction d'agrégation paramétrée (e.g., une somme pondérée, une intégrale de Choquet) mais que l'instance des paramètres permettant de représenter au mieux ses préférences n'est pas connue précisément. Dans ce cadre, nous avons proposé

d'une part des algorithmes pour la détermination de toutes les solutions potentiellement optimales, et d'autre part des procédures de résolution interactives permettant de déterminer efficacement une solution optimale (ou presque optimale) aux yeux du décideur.

Pour la recherche dans un graphe d'états multicritère, nous avons tout d'abord proposé des algorithmes de programmation dynamique pour la détermination des solutions potentiellement optimales lorsque les préférences du décideur sont représentables par une somme pondérée ou une intégrale de Choquet. Ces algorithmes sont des extensions de l'algorithme de MOA* (e.g., Mandow and Pérez de la Cruz [2005b]) utilisant des règles d'élagage exploitant le modèle de préférence imprécisément connu. Nous avons ensuite proposé une nouvelle approche pour la recherche fondée sur les préférences consistant à éliciter incrémentalement les paramètres du modèle décisionnel en cours de résolution. Nous avons proposé, implémenté et testé différentes stratégies de génération de questions permettant d'obtenir des compromis différents entre les deux objectifs conflictuels suivants : minimiser le nombre de questions et maximiser la qualité de la recommandation. Les tests ont montré que, pour ce problème d'optimisation multicritère, réaliser l'élicitation en cours de résolution donne de meilleurs résultats que la méthode en deux phases consistant tout d'abord à engendrer l'ensemble des solutions Pareto-optimales (avec MOA*) puis à appliquer des méthodes d'élicitation incrémentale standards sur cet ensemble. Par ailleurs, nous avons constaté que le nombre de questions engendrées par nos procédures reste relativement faible malgré la nature combinatoire du problème.

En ce qui concerne les problèmes d'arbres couvrants multicritères, nous avons tout d'abord proposé un algorithme glouton pour la détermination des arbres couvrants potentiellement optimaux lorsque les préférences du décideur sont représentables par une somme pondérée imprécisément connue. Cet algorithme est une généralisation multicritère de l'algorithme de Prim (e.g., [Prim, 1957]) utilisant les cocycles des arbres couvrants partiels pour décomposer l'espace des paramètres admissibles en régions caractérisant toutes les solutions potentiellement optimales. Cet algorithme peut en particulier être utilisé pour déterminer l'ensemble des solutions situées sur les bords de l'enveloppe convexe des solutions réalisables (solutions dite "supportées"). Cet algorithme a ensuite été sophistiqué pour obtenir un algorithme glouton interactif de complexité polynomiale, mêlant élicitation incrémentale et résolution, qui s'est avéré particulièrement efficace en pratique. Soulignons que, pour ce problème d'optimisation, il a récemment été montré que les points extrêmes de l'enveloppe convexe des solutions réalisables sont en nombre polynomial (à condition que le nombre de critères soit considéré comme une donnée fixée du problème) [Seipp, 2013] ; ce résultat pourrait probablement être exploité pour mettre en œuvre de nouvelles procédures incrémentales se concentrant uniquement sur ces points extrêmes.

Pour conclure, nous avons vu que la décision par élicitation incrémentale permet de concevoir des procédures de recommandation personnalisée relativement efficaces dans le cadre des problèmes de décision multicritère (chapitres 2 et 3). Dans les chapitres suivants, nous allons étudier le potentiel de cette approche pour résoudre des problèmes de décision collective (chapitre 4) et des problèmes de décision séquentielle dans le risque (chapitre 5).

Chapitre 4

Méthodes incrémentales pour la décision collective avec préférences incomplètes

Résumé

Ce chapitre présente nos travaux publiés dans [Benabbou et al., 2016b, Benabbou and Perny, 2016] concernant la problématique de vote en présence de préférences imprécisément connues par le système. Dans ce contexte, il s'agit de parvenir à déterminer les gagnants de l'élection en limitant le plus possible le nombre d'interactions avec les agents. Pour ce faire, nous adoptons ici l'approche d'élicitation incrémentale fondée sur la minimisation de regrets.

Dans un premier temps, nous étudions le potentiel de cette approche dans le cadre d'un problème de vote avec la méthode de Borda où les alternatives possibles sont décrites par plusieurs attributs et données explicitement. Nous montrons comment une représentation compacte des préférences des agents peut être exploitée dans ce cadre, pour définir des procédures de recommandation relativement efficaces en pratique, à la fois en terme de temps de calcul et de nombre de questions posées.

Dans un second temps, nous proposons une approche de résolution efficace pour le vote par approbation sur domaine combinatoire, illustrée sur le problème de sac à dos multi-agents. Dans ce contexte, il n'est pas envisageable d'énumérer toutes les solutions possibles pour identifier les gagnants de l'élection. C'est pourquoi, nous proposons plutôt de combiner la recherche et l'élicitation incrémentale des préférences pour réduire la taille de l'espace de recherche ainsi que le nombre de questions nécessaires à la prise de décision.

Le vote est une procédure permettant à un groupe de personnes de prendre une décision collective, tout en conférant une certaine légitimité à la décision. Cette pratique est omniprésente de nos jours, notamment durant les élections politiques et académiques. Depuis relativement récemment, le vote connaît un engouement croissant en informatique, compte tenu de la possibilité offerte par les systèmes en ligne de mettre en œuvre des protocoles de vote pour la prise de décision collective (e.g., WHALE [1]). Lorsque les préférences des votants sont entièrement connues, une procédure d'agrégation des préférences peut être utilisée pour déterminer la meilleure alternative pour le groupe d'individus (e.g., [Rossi et al., 2011]). Cependant, dans de nombreuses situations réelles, il est difficile de connaître précisément les préférences de chaque votant, car obtenir ces informations a généralement un coût économique et cognitif important. Dans ces situations, il est indispensable de bénéficier d'outils permettant de raisonner avec des préférences partielles. Cette observation a motivé de nombreuses contributions sur la problématique du vote avec préférences incomplètes. Dans cette ligne de recherche, les questions soulevées concernent notamment la détermination des gagnants possibles et nécessaires (e.g., [Konczak and Lang, 2005, Xia and Conitzer, 2011, Lang et al., 2012]) ainsi que la conception de méthodes interactives permettant de réduire l'ensemble des gagnants possibles jusqu'à pouvoir prendre une décision (e.g., [Kalech et al., 2010, Lu and Boutilier, 2011a,b, Ding and Lin, 2013, Dery et al., 2014, 2016]). Puisque l'acquisition de nouvelles données de préférences est une tâche généralement coûteuse, nous adoptons ici l'approche d'élicitation incrémentale fondée sur la minimisation de regrets pour identifier le gagnant de l'élection à moindre coût. Plus précisément, nous allons étudier le potentiel de cette approche tout d'abord dans un contexte de décision où l'ensemble des alternatives possibles est donné en extension (section 4.1), puis dans le cadre d'un problème sur domaine combinatoire (section 4.2).

4.1 Élicitation incrémentale avec la méthode de Borda sur domaine non combinatoire

Dans cette section, nous considérons une situation de décision où un ensemble $N = \{1, \ldots, n\}$ de votants (ou agents) doivent se mettre d'accord sur le choix d'une alternative dans l'ensemble $\mathcal{X}$ (e.g., candidats, options, objets). Chaque alternative x est ici décrite par plusieurs attributs (e.g., temps, prix, couleur) dont il faut tenir compte durant la prise de décision. Chaque alternative est donc associée à un vecteur $x = (x_1, \ldots, x_q)$, où x_j représente la performance de l'alternative x sur le critère $j \in \mathcal{Q}$. À titre d'exemple, la prise en compte simultanée de plusieurs critères dans la décision collective apparaît naturellement quand les différents agents doivent se mettre d'accord sur plusieurs questions à la fois (e.g., [Lang and Xia, 2009, Xia et al., 2011]). En effet, dans ces situations, les alternatives sont associées à des vecteurs de valeurs booléennes représentant les différentes décisions élémentaires. Plus généralement, le cas multicritère s'impose lorsque les alternatives à comparer sont évaluées sur plusieurs critères, comme par exemple le coût de la vie ou la température de la prochaine destination pour des vacances en famille. Dans ce cadre, nous supposons que les préférences de chaque agent $i \in N$ sont représentables par une

1. http ://strokes.imag.fr/whale4/

somme pondérée u^i de la forme suivante :

$$u^i(x) = \sum_{j=1}^{q} \omega_j^i x_j \tag{4.1}$$

où $\omega^i = (\omega_1^i, \ldots, \omega_q^i)$ est un jeu de poids positifs normalisé. Sous cette hypothèse, l'agent $i \in N$ préfère $x \in \mathcal{X}$ à $y \in \mathcal{X}$ si et seulement si $u^i(x) \geq u^i(y)$. Plus formellement, les préférences de l'agent $i \in N$ sont définies par la relation binaire $\succsim_i$ suivante :

$$x \succsim_i y \;\Leftrightarrow\; u^i(x) \geq u^i(y) \;\Leftrightarrow\; \sum_{j=1}^{q} \omega_j^i (x_j - y_j) \geq 0$$

Notons que, dans ce contexte, le profil de préférences $(\succsim_1, \ldots, \succsim_n)$ est complètement caractérisé par les n jeux de poids $\omega^1, \ldots, \omega^n$. Soulignons par ailleurs que notre définition des préférences autorise les agents à exprimer des indifférences entre les différentes alternatives du problème (contrairement à l'approche classique étudiée en théorie du vote).

Dans cette section, nous cherchons à déterminer l'alternative déclarée vainqueur de l'élection par la méthode de Borda (cf. Section 1.2.3). Étant donné un vecteur de jeux de poids $\omega = (\omega^1, \ldots, \omega^n)$, le score de Borda $\mathrm{B}(x, \omega)$ d'une alternative $x \in \mathcal{X}$ est définie par :

$$\mathrm{B}(x, \omega) = \sum_{i=1}^{n} \Big| \{ y \in \mathcal{X} : x \succ_i y \} \Big| = \sum_{i=1}^{n} \Big| \{ y \in \mathcal{X} : \sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i y_j \} \Big|$$

La relation $\succ_i$ représente la partie asymétrique de $\succsim_i$ formellement définie comme suit :

$$x \succ_i y \;\Leftrightarrow\; x \succsim_i y \text{ et } \neg(y \succsim_i x)$$

où $\neg$ représente l'opérateur de négation logique. Par définition, la meilleure alternative au sens de la méthode de Borda est celle qui maximise le score de Borda. Notons que si les vecteurs de poids $\omega^i, i \in N$, ne modélisent aucune indifférence (i.e. $u^i(x) \neq u^i(y)$ pour toutes alternatives $x, y \in \mathcal{X}$ distinctes), alors notre définition du score de Borda est la même que celle utilisée en théorie du choix social.

Lorsque les préférences des agents ne sont pas connues précisément, nous devons raisonner avec des préférences partiellement définies. Plus précisément, nous considérons ici un vecteur d'ensembles $\Omega = (\Omega^1, \ldots, \Omega^n)$, où chaque composante Ω^i représente l'ensemble des jeux de poids ω^i compatibles avec $\mathcal{P}^i$ l'ensemble des informations disponibles sur les préférences de l'agent $i \in N$. Supposons que l'ensemble $\mathcal{P}^i$ ne contient que des paires (a, b) représentant la préférence "a meilleure que b". Comme chacune de ces paires impose uniquement la contrainte linéaire $\sum_{j=1}^{q} \omega_j^i (a_j - b_j) \geq 0$ sur l'espace des jeux de poids admissibles, alors chaque ensemble Ω^i forme ici un polyèdre convexe.

Dans ce contexte, un *gagnant possible* au sens de Borda est une alternative $x \in \mathcal{X}$ telle que, pour au moins une instance $\omega \in \Omega$, l'inégalité $\mathrm{B}(x, \omega) \geq \mathrm{B}(y, \omega)$ est satisfaite pour tout $y \in \mathcal{X}$. Similairement, une

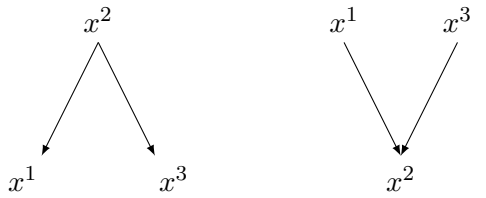
alternative $x \in \mathcal{X}$ est un *gagnant nécessaire* au sens de Borda si et seulement si $\mathrm{B}(x,\omega) \geq \mathrm{B}(y,\omega)$ pour tout $\omega \in \Omega$ et tout $y \in \mathcal{X}$. Si un gagnant nécessaire existe, alors cette alternative peut être directement recommandée au groupe d'individus car celle-ci maximise le score de Borda dans tous les scénarios possibles. Dans le cas contraire, il s'agit pour le système de collecter davantage d'informations sur les préférences des agents pour pouvoir formuler une recommandation.

Afin de limiter le nombre de questions posées aux agents, nous adoptons ici l'approche d'élicitation incrémentale fondée sur le critère Minimax Regret (présentée en Section 1.4.2) Ainsi, nous travailler ci-après avec les notions de *pairwise max regret* (PMR), de *max regret* (MR) et de *minimax regret* (mMR), comme proposé dans d'autres travaux (e.g., [Lu and Boutilier, 2011b]) :

$$\begin{aligned}
\mathrm{PMR}(x,y,\Omega) &= \max_{\omega\in\Omega}\Big\{\mathrm{B}(y,\omega)-\mathrm{B}(x,\omega)\Big\}\\
\mathrm{MR}(x,\mathcal{X},\Omega) &= \max_{y\in\mathcal{X}}\mathrm{PMR}(x,y,\Omega)\\
\mathrm{mMR}(\mathcal{X},\Omega) &= \min_{x\in\mathcal{X}}\mathrm{MR}(x,\mathcal{X},\Omega)
\end{aligned}$$

Le $\mathrm{PMR}(x,y,\Omega)$ de $x \in \mathcal{X}$ par rapport à $y \in \mathcal{X}$ est le plus grand écart de scores possible entre l'alternative y et l'alternative x. La quantité $\mathrm{MR}(x,\mathcal{X},\Omega)$ représente quant à elle la plus grande différence de scores possible entre le vainqueur de l'élection et l'alternative x. Les meilleures alternatives pour le critère de décision Minimax Regret sont celles qui minimisent la valeur MR, la quantité $\mathrm{mMR}(\mathcal{X},\Omega)$ représentant alors cette plus petite valeur. Ces alternatives sont en quelque sorte des approximations "robustes" du gagnant de l'élection car la recommandation de ces alternatives permet de minimiser l'erreur commise dans le pire scénario possible. Ainsi, choisir parmi ces alternatives constitue la décision la plus sûre au vu de l'imprécision qui entoure les préférences des agents. Soulignons toutefois que les alternatives optimales pour ce critère de décision ne sont pas toujours des gagnants possibles, comme le montre l'exemple suivant :

Exemple 28. *Considérons un problème de vote dans lequel $n = 5k$ agents doivent se mettre d'accord sur la sélection d'une alternative de l'ensemble $\mathcal{X} = \{x^1, x^2, x^3\}$, où k est un entier strictement positif. Supposons que les informations disponibles sur les préférences de ces n agents nous offrent la possibilité de les diviser en deux groupes, de sorte que le profil de préférences partiel représentant cette situation de vote soit le suivant :*



I. $2k$ agents *II. $3k$ agents*

En résumé, la seule information dont nous disposons est que x^2 est strictement meilleure que x^1 et x^3 aux yeux des $2k$ agents du groupe I, alors que les agents du groupe II considèrent que x^1 et x^3 sont toutes

deux strictement meilleures que x^2. Notons que le score de Borda de l'alternative x^2 est forcément égal à $4k$, quelque soit la complétion du profil de préférences considérée. En effet, l'alternative x^2 reçoit :

- *deux points par agent du groupe I car ces derniers considèrent que x^2 bat strictement x^1 et x^3.*
- *zéro point par personne du groupe II car celles-ci estiment que x^2 est le plus mauvais choix possible.*

Notons par ailleurs que la meilleure alternative de chaque agent du groupe II fait forcément partie de l'ensemble $\{x^1, x^3\}$. De ce fait, pour toute complétion possible du profil de préférences, il existe $j \in \{1, 3\}$ tel que x^j est strictement préférée aux deux autres alternatives du problème par au moins $3k/2$ agents du groupe II. Comme les autres agents de ce groupe trouvent que x^j est strictement meilleure que x^2, alors x^j reçoit au moins $2 \times 3k/2 + 3k = 6k$ points des membres du groupe II. Ainsi, l'alternative x^2 de score de $4k$ ne fait pas partie des gagnants possibles par la méthode de Borda. Pourtant, l'alternative x^2 est l'unique solution optimale au sens du critère de décision Minimax Regret. Pour le voir, nous reportons dans la case (k, l) du tableau suivant la valeur $\mathrm{PMR}(x^k, x^l, \Omega)$ *:*

k \ l	1	2	3
1	0	$4k - 3k = k$	$8k - 3k = 5k$
2	$8k - 4k = 4k$	0	$8k - 4k = 4k$
3	$8k - 3k = 5k$	$4k - 3k = k$	0

De ce tableau, nous pouvons déduire $\mathrm{MR}(x^1, \{x^1, x^2, x^3\}, \Omega) = 5k$, $\mathrm{MR}(x^2, \{x^1, x^2, x^3\}, \Omega) = 4k$ *et* $\mathrm{MR}(x^3, \{x^1, x^2, x^3\}, \Omega) = 5k$, *ce qui montre que x^2 est l'unique solution optimale pour le critère de décision Minimax Regret.*

Observons par ailleurs que si $\mathrm{mMR}(\mathcal{X}, \Omega) = 0$, alors une alternative optimale au sens du Minimax Regret représente en réalité un gagnant nécessaire de l'élection. Ainsi, il n'est pas nécessaire de poser des questions aux agents pour pouvoir prendre une décision dans ce cas. Dans le cas contraire, il est possible que la valeur $\mathrm{mMR}(\mathcal{X}, \Omega)$ soit bien trop élevée pour certifier que le critère Minimax Regret sélectionne des alternatives de qualité. Dans ces situations, une méthode d'élicitation incrémentale peut être envisagée, posant des questions aux agents de manière itérative pour réduire progressivement l'ensemble Ω, tant que $\mathrm{mMR}(\mathcal{X}, \Omega) > \delta$, où $\delta \geq 0$ est un seuil de tolérance représentant une borne sur l'erreur acceptable. Pour pouvoir mettre en œuvre une telle procédure itérative, il convient de proposer des méthodes efficaces pour le calcul des valeurs $\mathrm{mMR}(\Omega, \mathcal{X})$ à chaque itération. Comme nous l'avons déjà souligné, la résolution des problèmes d'optimisation PMR constitue la pierre angulaire de ce calcul. En effet, une fois que la valeur $\mathrm{PMR}(x, y, \Omega)$ est connue pour toutes les paires $(x, y) \in \mathcal{X}^2$, nous pouvons déduire la quantité $\mathrm{MR}(x, \mathcal{X}, \Omega)$ pour toute alternative $x \in \mathcal{X}$, ce qui permet de dériver la valeur $\mathrm{mMR}(\Omega, \mathcal{X})$ à la fin.

Dans cette sous-section, nous commençons donc par étudier la problématique du calcul efficace de la valeur $\mathrm{PMR}(x, y, \Omega)$ pour une paire $(x, y) \in \mathcal{X}^2$ quelconque. Puis, nous nous intéresserons à la conception de stratégies d'élicitation permettant de recommander une alternative de qualité en limitant le nombre de questions posées aux agents.

4.1.1 Travaux connexes

Avant de décrire notre méthode pour le calcul de $\mathrm{PMR}(x, y, \Omega)$, nous présentons des travaux récents sur l'élicitation incrémentale du gagnant au sens de Borda pour des problèmes de vote où les alternatives ne sont pas multivaluées [Lu and Boutilier, 2011b]. Dans ce contexte, la connaissance du système sur la relation de préférence $\succ_i$ de l'agent $i \in N$ se résume à l'ensemble $\mathcal{P}^i_>$ des paires (a, b) représentant la préférence "a strictement meilleure que b" formulée explicitement par l'agent i. Il s'agit alors de raisonner avec l'ensemble $\mathcal{R}^i$ de toutes les relations binaires complètes $\rhd_i$ qui vérifient :

$$(x, y) \in \mathcal{P}^i \;\Rightarrow\; x \rhd_i y$$

Les définitions de regrets associées sont les suivantes :

$$\begin{aligned}
\mathrm{PMR}(x, y, \mathcal{R}) &= \max_{\rhd \in \mathcal{R}} \Big\{ \mathrm{B}(y, \rhd) - \mathrm{B}(x, \rhd) \Big\} \\
\mathrm{MR}(x, \mathcal{X}, \mathcal{R}) &= \max_{y \in \mathcal{X}} \mathrm{PMR}(x, y, \mathcal{R}) \\
\mathrm{mMR}(\mathcal{X}, \mathcal{R}) &= \min_{x \in \mathcal{X}} \mathrm{MR}(x, \mathcal{R})
\end{aligned}$$

où $\mathcal{R}$ représente le produit cartésien $\mathcal{R} = \mathcal{R}^1 \times \ldots \times \mathcal{R}^n$ et $\mathrm{B}(w, \rhd) = \sum_{i \in N} |\{z \in \mathcal{X} : w \rhd_i z\}|$ est le score de Borda de l'alternative $w \in \mathcal{X}$ associé au profil $\rhd = (\rhd^1, \ldots, \rhd^n)$ (cf. définition 24). Pour résoudre efficacement le problème $\mathrm{PMR}(x, y, \mathcal{R})$ pour une paire $(x, y) \in \mathcal{X}^2$ quelconque, les auteurs de ces travaux exploitent le résultat suivant :

$$\begin{aligned}
\mathrm{PMR}(x, y, \mathcal{R}) &= \max_{\rhd \in \mathcal{R}} \Big\{ \mathrm{B}(y, \rhd) - \mathrm{B}(x, \rhd) \Big\} \\
&= \max_{\rhd \in \mathcal{R}} \sum_{i=1}^{n} \Big\{ |\{z \in \mathcal{X} : y \rhd_i z\}| - |\{z \in \mathcal{X} : x \rhd_i z\}| \Big\} \\
&= \sum_{i=1}^{n} \max_{\rhd_i \in \mathcal{R}^i} \Big\{ |\{z \in \mathcal{X} : y \rhd_i z\}| - |\{z \in \mathcal{X} : x \rhd_i z\}| \Big\}
\end{aligned}$$

Cette formulation permet en effet de décomposer le problème d'optimisation $\mathrm{PMR}(x, y, \mathcal{R})$ en une série de sous-problèmes plus simples. Plus précisément, pour chaque agent $i \in N$, il s'agit de déterminer sa "contribution" à la valeur $\mathrm{PMR}(x, y, \mathcal{R})$, notée $\mathrm{PMR}^i(x, y, \mathcal{R})$ et définie par :

$$\mathrm{PMR}^i(x, y, \mathcal{R}) = \max_{\rhd_i \in \mathcal{R}^i} \Big\{ |\{z \in \mathcal{X} : y \rhd_i z\}| - |\{z \in \mathcal{X} : x \rhd_i z\}| \Big\}$$

La valeur $\mathrm{PMR}^i(x, y, \mathcal{R})$ correspond à la différence maximale entre le nombre d'alternatives strictement battues par y et le nombre d'alternatives strictement battues par x (au sens des préférences subjectives de l'agent i). Finalement, pour calculer la valeur $\mathrm{PMR}^i(x, y, \mathcal{R})$ de manière efficace, il suffit de distinguer les trois cas suivants :

- *Cas où* $(x, y) \in \mathcal{P}^i_>$ *:* dans ce cas, nous avons : $\forall \rhd_i \in \mathcal{R}^i, \forall z \in \mathcal{X}, y \rhd_i z \Rightarrow x \rhd_i z$. Ceci nous

permet de déduire le résultat suivant :

$$\begin{aligned}\mathrm{PMR}^i(x,y,\mathcal{R}) &= \max_{\rhd_i\in\mathcal{R}^i}\Big\{-|\{z\in\mathcal{X} : x\rhd_i z \text{ et } \neg(y\rhd_i z)\}|\Big\}\\ &= \max_{\rhd_i\in\mathcal{R}^i}\Big\{-|\{z\in\mathcal{X} : x\rhd_i z\rhd_i y\}|\Big\}\\ &= -\min_{\rhd_i\in\mathcal{R}^i}\Big\{|\{z\in\mathcal{X} : x\rhd_i z\rhd_i y\}|\Big\}\\ &= -|\{z\in\mathcal{X} : (x,z)\in\mathcal{P}^i_> \text{ et } (z,y)\in\mathcal{P}^i_>\}|\end{aligned}$$

- *Cas où* $(y,x)\in\mathcal{P}^i_>$ *:* nous avons ici : $\forall\rhd_i\in\mathcal{R}^i, \forall z\in\mathcal{X}, x\rhd_i z\Rightarrow y\rhd_i z$. De ce fait :

$$\begin{aligned}\mathrm{PMR}^i(x,y,\mathcal{R}) &= \max_{\rhd_i\in\mathcal{R}^i}\Big\{|\{z\in\mathcal{X} : y\rhd_i z \text{ et } \neg(x\rhd_i z)\}|\Big\}\\ &= \max_{\rhd_i\in\mathcal{R}^i}\Big\{|\{z\in\mathcal{X} : y\rhd_i z\rhd_i x\}|\Big\}\\ &= |\{z\in\mathcal{X} : (z,y)\notin\mathcal{P}^i_> \text{ et } (x,z)\notin\mathcal{P}^i_>\}|\end{aligned}$$

- *Cas où* $(y,x)\notin\mathcal{P}^i_>$ *et* $(x,y)\notin\mathcal{P}^i_>$ *:* nous venons de voir que $\mathrm{PMR}^i(x,y,\mathcal{R})\geq 0$ lorsque $(y,z)\in\mathcal{P}^i_>$ et que $\mathrm{PMR}^i(x,y,\mathcal{R})\leq 0$ dans le cas où $(x,y)\in\mathcal{P}^i_>$. Par conséquent, dans la situation où $(y,x)\notin\mathcal{P}^i_>$ et $(x,y)\notin\mathcal{P}^i_>$, nous pouvons obtenir la valeur $\mathrm{PMR}^i(x,y,\mathcal{R})$ en nous restreignant aux relations $\rhd_i\in\mathcal{R}^i$ telles que $x\rhd_i y$. Dans cette situation, la valeur $\mathrm{PMR}^i(x,y,\mathcal{R})$ peut donc être calculée exactement comme dans le cas $(y,x)\in\mathcal{P}^i_>$.

Ainsi, dans tous les cas possibles, le problème d'optimisation $\mathrm{PMR}^i(x,y,\mathcal{R})$ peut être résolu en $O(|\mathcal{X}|)$ en comptant le nombre d'alternatives $z\in\mathcal{X}$ vérifiant des conditions très simples sur les préférences déjà formulées par l'agent i. Dans la sous-section suivante, nous montrons comment étendre cette approche aux situations de vote où les alternatives sont multivaluées et que les préférences de chaque agent $i\in N$ sont modélisées à l'aide d'une somme pondérée u^i de paramètres ω^i (cf. équation (4.1)).

4.1.2 Calcul du Minimax Regret par programmation linéaire en nombres entiers

Nous nous intéressons maintenant au calcul de $\mathrm{PMR}(x,y,\Omega)$, étant donné un vecteur d'ensembles $\Omega=(\Omega^1,\ldots,\Omega^n)$, où chaque ensemble Ω^i représente l'imprécision subsistante sur le jeu de poids ω^i permettant de définir $\succsim_i$, la relation de préférence de l'agent i (cf. équation (4.1)). Comme dans la sous-section précédente, nous commençons par décomposer le problème $\mathrm{PMR}(x,y,\Omega)$:

$$\begin{aligned}\mathrm{PMR}(x,y,\Omega) &= \max_{\omega\in\Omega}\Big\{\mathrm{B}(y,\omega)-\mathrm{B}(x,\omega)\Big\}\\ &= \max_{\omega\in\Omega}\sum_{i=1}^{n}\Big\{|\{z\in\mathcal{X} : \sum_{j=1}^{q}\omega^i_j y_j > \sum_{j=1}^{q}\omega^i_j z_j\}| - |\{z\in\mathcal{X} : \sum_{j=1}^{q}\omega^i_j x_j > \sum_{j=1}^{q}\omega^i_j z_j\}|\Big\}\\ &= \sum_{i=1}^{n}\max_{\omega^i\in\Omega^i}\Big\{|\{z\in\mathcal{X} : \sum_{j=1}^{q}\omega^i_j y_j > \sum_{j=1}^{q}\omega^i_j z_j\}| - |\{z\in\mathcal{X} : \sum_{j=1}^{q}\omega^i_j x_j > \sum_{j=1}^{q}\omega^i_j z_j\}|\Big\} \quad (4.2)\end{aligned}$$

Il s'agit donc de résoudre, pour chaque agent $i \in N$, le problème d'optimisation suivant :

$$\mathrm{PMR}^i(x, y, \Omega^i) = \max_{\omega^i \in \Omega^i} \left\{ |\{z \in \mathcal{X} : \sum_{j=1}^{q} \omega_j^i y_j > \sum_{j=1}^{q} \omega_j^i z_j\}| - |\{z \in \mathcal{X} : \sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j\}| \right\}$$

Dans notre contexte multicritère, la résolution de ce problème est plus complexe car celle-ci nécessite de gérer la structure multicritère du domaine. En effet, dans ce contexte, la connaissance du système sur les préférences des agents ne se résume pas aux ensembles $\mathcal{P}^i, i \in N$, mais aux ensembles $\Omega^i, i \in N$, qui peuvent permettre de dériver des préférences non encore formulées par les agents. C'est pourquoi, pour déterminer la valeur $\mathrm{PMR}^i(x, y, \Omega^i)$ pour un agent $i \in N$ donné, nous proposons de travailler avec les relations binaires suivantes :

- Une alternative $z \in \mathcal{X}$ est dite *nécessairement préférée* à une alternative $z' \in \mathcal{X}$ par l'agent i, noté $z \succsim_i^{Nec} z'$, si et seulement si : $\forall \omega^i \in \Omega^i, \sum_{j=1}^{q} \omega_j^i z_j \geq \sum_{j=1}^{q} \omega_j^i z'_j$.
- Une alternative $z \in \mathcal{X}$ est dite *nécessairement strictement préférée* à une alternative $z' \in \mathcal{X}$ par l'agent i, noté $z \succ_i^{Nec} z'$, si et seulement si : $\forall \omega^i \in \Omega^i, \sum_{j=1}^{q} \omega_j^i z_j > \sum_{j=1}^{q} \omega_j^i z'_j$.

Notons que vérifier si $z \succsim_i^{Nec} z'$ (resp. $z \succ_i^{Nec} z'$) pour une paire $(z, z') \in \mathcal{X}^2$ revient à tester si la quantité $\min_{\omega^i \in \Omega^i} \{\sum_{j=1}^{q} \omega_j^i z_j - \sum_{j=1}^{q} \omega_j^i z'_j\}$ est positive (resp. strictement positive). Cette quantité correspond à la valeur optimale du programme linéaire suivant :

$$\min_{\omega^i} \quad \sum_{j=1}^{q} \omega_j^i z_j - \sum_{j=1}^{q} \omega_j^i z'_j \tag{4.3}$$

$$s.c. \quad \sum_{j=1}^{q} \omega_j^i = 1 \tag{4.4}$$

$$\sum_{j=1}^{q} \omega_j^i a_j - \sum_{j=1}^{q} \omega_j^i b_j \geq 0, \, \forall (a, b) \in \mathcal{P}_{\geq}^i \tag{4.5}$$

$$\sum_{j=1}^{q} \omega_j^i a_j - \sum_{j=1}^{q} \omega_j^i b_j \geq \xi, \, \forall (a, b) \in \mathcal{P}_{>}^i \tag{4.6}$$

$$\omega_j^i \geq 0, \; \forall j \in \mathcal{Q} \tag{4.7}$$

où ξ est une constante positive arbitrairement petite servant à modéliser les inégalités strictes. Toutes les contraintes de ce programme permettent de décrire l'ensemble Ω^i des jeux de poids ω^i admissibles. Plus précisément, les contraintes (4.4) et (4.7) permettent de garantir que ω^i est un vecteur normalisé de poids positifs tandis que les équations (4.5) et (4.6) assurent que le jeu de poids ω^i satisfait les contraintes induites respectivement par $\mathcal{P}_{\geq}^i$ l'ensemble des préférences faibles et $\mathcal{P}_{>}^i$ l'ensemble des préférences strictes formulées par l'agent i. Nous distinguons ensuite les trois cas suivants :

- *Cas où $x \succsim_i^{Nec} y$:* nous avons $\sum_{j=1}^{q} \omega_j^i x_j \geq \sum_{j=1}^{q} \omega_j^i y_j$ pour tout $\omega^i \in \Omega^i$. Par conséquent, pour tout $\omega^i \in \Omega^i$ et tout $z \in \mathcal{X}$, nous avons : $\sum_{j=1}^{q} \omega_j^i y_j > \sum_{j=1}^{q} \omega_j^i z_j \Rightarrow \sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j$.

Ainsi, nous pouvons réécrire le problème d'optimisation de la manière suivante :

$$\begin{aligned}\mathrm{PMR}^i(x,y,\Omega^i) &= \max_{\omega^i\in\Omega^i}\Big\{-|\{z\in\mathcal{X}:\sum_{j=1}^q\omega_j^i x_j>\sum_{j=1}^q\omega_j^i z_j \text{ et } \neg\Big(\sum_{j=1}^q\omega_j^i y_j>\sum_{j=1}^q\omega_j^i z_j\Big)\}|\Big\}\\ &= \max_{\omega^i\in\Omega^i}\Big\{-|\{z\in\mathcal{X}:\sum_{j=1}^q\omega_j^i x_j>\sum_{j=1}^q\omega_j^i z_j\geq\sum_{j=1}^q\omega_j^i y_j\}|\Big\}\\ &= -\min_{\omega^i\in\Omega^i}\Big\{|\{z\in\mathcal{X}:\sum_{j=1}^q\omega_j^i x_j>\sum_{j=1}^q\omega_j^i z_j\geq\sum_{j=1}^q\omega_j^i y_j\}|\Big\}\end{aligned}$$

- *Cas où* $\neg(x\succsim_i^{Nec}y)\wedge(y\succsim_i^{Nec}x)$ *:* dans ce cas, nous avons $\sum_{j=1}^q\omega_j^i y_j\geq\sum_{j=1}^q\omega_j^i x_j$ pour tout $\omega^i\in\Omega^i$. De façon similaire au cas précédent, nous en déduisons :

$$\begin{aligned}\mathrm{PMR}^i(x,y,\Omega^i) &= \max_{\omega^i\in\Omega^i}\Big\{|\{z\in\mathcal{X}:\sum_{j=1}^q\omega_j^i y_j>\sum_{j=1}^q\omega_j^i z_j \text{ et } \neg\Big(\sum_{j=1}^q\omega_j^i x_j>\sum_{j=1}^q\omega_j^i z_j\Big)\}|\Big\}\\ &= \max_{\omega^i\in\Omega^i}\Big\{|\{z\in\mathcal{X}:\sum_{j=1}^q\omega_j^i y_j>\sum_{j=1}^q\omega_j^i z_j\geq\sum_{j=1}^q\omega_j^i x_j\}|\Big\}\end{aligned}$$

- *Cas où* $\neg(x\succsim_i^{Nec}y)\wedge\neg(y\succsim_i^{Nec}x)$ *:* dans ce cas, comme nous avons $\neg(x\succsim^{Nec}y)$, alors il existe au moins un jeu de poids $\omega^i\in\Omega^i$ tel que $\sum_{j=1}^q\omega_j^i y_j>\sum_{j=1}^q\omega_j^i x_j$. Pour de tels jeux de poids ω^i, nous avons vu que la fonction à maximiser se réécrit $|\{z\in\mathcal{X}:\sum_{j=1}^q\omega_j^i y_j>\sum_{j=1}^q\omega_j^i z_j\geq\sum_{j=1}^q\omega_j^i x_j\}|$, qui est une valeur positive. Pour les autres jeux de poids de l'ensemble Ω^i, nous avons vu que la fonction à maximiser est $-|\{z\in\mathcal{X}:\sum_{j=1}^q\omega_j^i x_j>\sum_{j=1}^q\omega_j^i z_j\geq\sum_{j=1}^q\omega_j^i y_j\}|$, qui est négative. Par conséquent, la valeur $\mathrm{PMR}^i(x,y,\Omega^i)$ peut être obtenue avec un jeu de poids $\omega^i\in\Omega^i$ tel que $\sum_{j=1}^q\omega_j^i y_j>\sum_{j=1}^q\omega_j^i x_j$, et donc cette valeur peut être calculée comme dans le cas précédent.

La suite de cette sous-section est consacrée au calcul de $\mathrm{PMR}^i(x,y,\Omega^i)$ dans les trois scénarios possibles (présentés ci-dessus) pour un agent i donné. Avant de présenter notre méthode de résolution, nous introduisons les ensembles suivants pour tout $a\in\{x,y\}$:

- $P^a=\{z\in\mathcal{X}\setminus\{a\}:z\succsim_i^{Nec}a\}$: l'ensemble des alternatives nécessairement préférées à a.
- $B^a=\{z\in\mathcal{X}:a\succ_i^{Nec}z\}$: l'ensemble des alternatives nécessairement strictement battues par a.
- $I^a=\mathcal{X}\setminus(\{a\}\cup P^a\cup B^a)$: l'ensemble des alternatives ni nécessairement préférées à a ni nécessairement strictement battues par a.

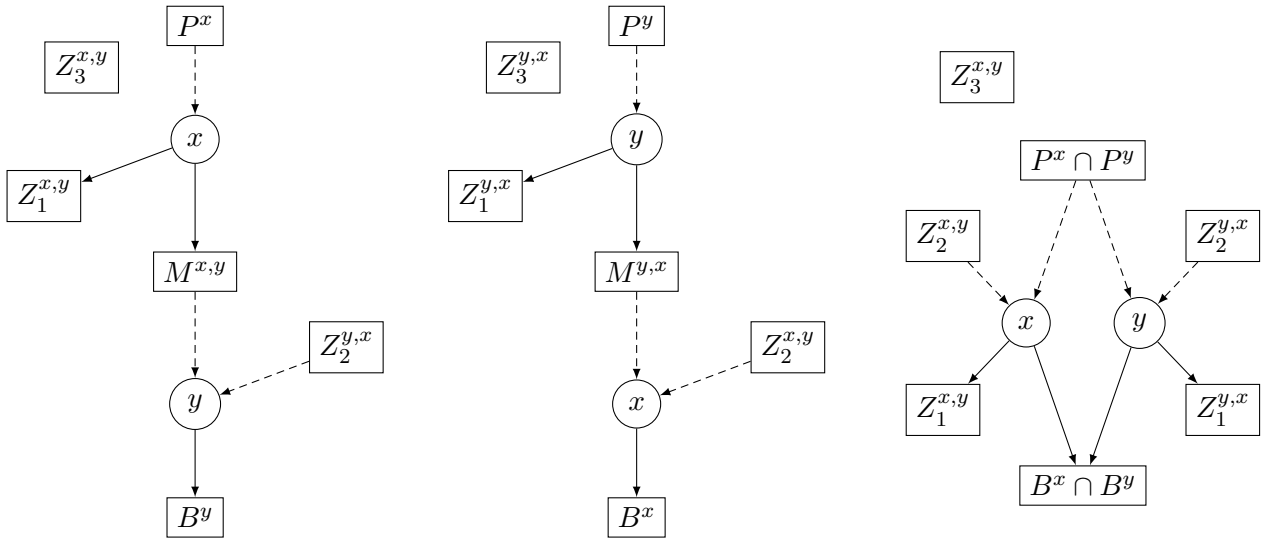
et pour toute paire d'alternatives $(a,b)\in\{(x,y),(y,x)\}$:

- $M^{a,b}=B^a\cap P^b$: l'ensemble des alternatives nécessairement strictement battues par a mais aussi nécessairement préférées à b.
- $Z_1^{a,b}=B^a\cap I^b$: l'ensemble des alternatives qui sont d'une part nécessairement strictement battues par a et d'autre part ni nécessairement préférées à b ni nécessairement strictement battues par b.
- $Z_2^{b,a}=P^b\cap I^a$: l'ensemble des alternatives qui sont d'une part nécessairement préférées à b et

d'autre part ni nécessairement préférées à a ni nécessairement strictement battues par a.

- $Z_3^{a,b} = I^a \cap I^b$: l'ensemble des alternatives qui sont ni nécessairement strictement battues par a, ni nécessairement préférées à a, ni nécessairement strictement battues par b, ni nécessairement préférées à b.

Ces ensembles, qui partitionnent l'ensemble $\mathcal{X}$, vont être utiles pour déterminer la valeur $\mathrm{PMR}^i(x, y, \Omega^i)$ selon les différents cas énoncés précédemment. Pour une illustration graphique, le lecteur est invité à se référer à la figure 4.1. Dans la suite, afin de faciliter la lecture, nous écrirons Z_1, Z_2 et Z_3 (sans les exposants) lorsque le cas considéré est clairement identifié.



(a) $x \succsim_i^{Nec} y$ (b) $\neg(x \succsim_i^{Nec} y) \wedge (y \succsim_i^{Nec} x)$ (c) $\neg(x \succsim_i^{Nec} y) \wedge \neg(y \succsim_i^{Nec} x)$

FIGURE 4.1 – Partition de l'ensemble $\mathcal{X}$ selon les valeurs de $x \succsim_i^{Nec} y$ et $y \succsim_i^{Nec} x$. Les arcs continus (resp. tiretés) représentent des préférences nécessairement strictes (resp. faibles).

• *Cas où $x \succsim_i^{Nec} y$ (cf. figure 4.1a) :* il s'agit pour nous de calculer $\mathrm{PMR}^i(x, y, \Omega^i)$. Rappelons que, dans ce cas, nous avons $\mathrm{PMR}^i(x, y, \Omega^i) = -\min_{\omega^i \in \Omega^i} |\{z \in \mathcal{X} : \sum_{j=1}^q \omega_j^i x_j > \sum_{j=1}^q \omega_j^i z_j \geq \sum_{j=1}^q \omega_j^i y_j\}|$. Nous souhaitons donc déterminer un jeu de poids $\omega^i \in \Omega^i$ permettant de minimiser le nombre d'alternatives $z \in \mathcal{X}$ vérifiant la propriété suivante :

$$\sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j \geq \sum_{j=1}^{q} \omega_j^i y_j$$

Tout d'abord, remarquons que, pour tout jeu de poids $\omega^i \in \Omega^i$, aucune alternative $z \in P^x \cup B^y$ ne peut satisfaire cette propriété (par définition de P^x et B^y). En revanche, pour tout jeu de poids $\omega^i \in \Omega^i$, toutes les alternatives $z \in M^{x,y}$ vérifient cette propriété (par définition de $M^{x,y}$). Par conséquent :

$$\mathrm{PMR}^i(x, y, \Omega^i) = -|M^{x,y}| - \min_{\omega^i \in \Omega^i} |\{z \in Z_1 \cup Z_2 \cup Z_3 \cup \{y\} : \sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j \geq \sum_{j=1}^{q} \omega_j^i y_j\}|$$

Il suffit donc de calculer $\min_{\omega^i \in \Omega^i} |\{z \in Z_1 \cup Z_2 \cup Z_3 \cup \{y\} : \sum_{j=1}^q \omega_j^i x_j > \sum_{j=1}^q \omega_j^i z_j \geq \sum_{j=1}^q \omega_j^i y_j\}|$ pour pouvoir déterminer la valeur $\mathrm{PMR}^i(x, y, \Omega^i)$. Pour ce faire, nous introduisons le programme linéaire en nombres entiers suivant (nommé $\mathrm{PLNE}_{x,y}$) :

$$
\begin{aligned}
\min_{\omega^i} \quad & b^y + \sum_{z \in Z_1} b_1^z + \sum_{z \in Z_2} b_2^z + \sum_{z \in Z_3} b_3^z \\
\text{s.c.} \quad & \sum_{j=1}^q \omega_j^i = 1 && (4.8)\\
& \sum_{j=1}^q \omega_j^i a_j - \sum_{j=1}^q \omega_j^i b_j \geq 0,\ \forall (a,b) \in \mathcal{P}_{\geq}^i && (4.9)\\
& \sum_{j=1}^q \omega_j^i a_j - \sum_{j=1}^q \omega_j^i b_j \geq \xi,\ \forall (a,b) \in \mathcal{P}_{>}^i && (4.10)\\
& \sum_{j=1}^q \omega_j^i y_j - \sum_{j=1}^q \omega_j^i x_j + M b^y \geq 0 && (4.11)\\
& \sum_{j=1}^q \omega_j^i y_j - \sum_{j=1}^q \omega_j^i z_j + M b_1^z \geq \xi,\ \forall z \in Z_1 \cup Z_3 && (4.12)\\
& \sum_{j=1}^q \omega_j^i z_j - \sum_{j=1}^q \omega_j^i x_j + M b_2^z \geq 0,\ \forall z \in Z_2 \cup Z_3 && (4.13)\\
& b_3^z \geq b_1^z + b_2^z - 1,\ \forall z \in Z_3 && (4.14)\\
& \omega_j^i \geq 0,\ \forall j \in \{1, \ldots, q\} && (4.15)\\
& b^y \in \{0,1\};\ b_1^z \in \{0,1\},\ \forall z \in Z_1 \cup Z_3 && (4.16)\\
& b_2^z \in \{0,1\},\ \forall z \in Z_2 \cup Z_3 \\
& b_3^z \in \{0,1\},\ \forall z \in Z_3
\end{aligned}
$$

Dans ce programme, l'ensemble des variables est constitué d'une variable par composante du vecteur $\omega^i = (\omega_1^i, \ldots, \omega_q^i)$, d'une variable binaire b^y associée à l'alternative y, d'une variable binaire b_1^z pour chaque alternative $z \in Z_1 \cup Z_3$, d'une variable binaire b_2^z pour chaque alternative $z \in Z_2 \cup Z_3$, et d'une variable binaire b_3^z pour chaque alternative $z \in Z_3$; nous avons donc $q + |Z_1| + |Z_2| + 3|Z_3| + 1$ variables. Par ailleurs, M (resp. ξ) est une constante positive arbitrairement grande (resp. petite) permettant de modéliser des implications (resp. inégalités strictes). Enfin, les équations (4.8), (4.9), (4.10) et (4.15) contraignent le vecteur de poids ω^i à faire partie de l'ensemble Ω^i des jeux de poids admissibles. Cette formulation linéaire en nombres entiers est utile pour calculer la valeur $\mathrm{PMR}^i(x, y, \Omega^i)$, comme le montre la proposition suivante :

Proposition 33. *Si $x \succsim_i^{Nec} y$, alors $\mathrm{PMR}^i(x, y, \Omega^i) = -|M^{x,y}| - OPT$, où OPT représente la valeur optimale de $PLNE_{x,y}$.*

Démonstration. Montrons que $\min_{\omega^i \in \Omega^i} |\{z \in Z_1 \cup Z_2 \cup Z_3 \cup \{y\} : \sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j \geq \sum_{j=1}^{q} \omega_j^i y_j\}|$ correspond à la valeur optimale de $\text{PLNE}_{x,y}$. Pour ce faire, il suffit de montrer que, pour un jeu de poids ω^i donné, la fonction objectif de ce programme linéaire en nombres entiers compte le nombre d'alternatives $z \in Z_1 \cup Z_2 \cup Z_3 \cup \{y\}$ qui vérifient les inégalités suivantes :

$$\sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j \geq \sum_{j=1}^{q} \omega_j^i y_j \tag{4.17}$$

Dans ce programme, nous utilisons un ensemble de variables binaires b_1^z, b_2^z et b_3^z afin de représenter le fait que la condition (4.17) soit satisfaite pour les alternatives z dans Z_1, Z_2 et Z_3 respectivement. La variable binaire b^y modélise quant à elle le fait que la condition soit vraie pour l'alternative y. Puisque la fonction objectif de ce programme est la somme de toutes ces variables binaires, alors il suffit de prouver que chacune d'elles est instanciée à la valeur 1 si et seulement si l'alternative z associée à cette variable vérifie la propriété (4.17). Comme la fonction objectif doit être minimisée, alors ces variables sont mis à la valeur 0 sauf si des contraintes du programme les forcent à être égales à 1. Par conséquent, notre objectif est de montrer que ces variables sont contraintes de valoir 1 si et seulement si l'équation (4.17) est vérifiée par l'alternative z correspondante.

Pour toute alternative $z \in Z^1$, remarquons que la variable b_1^z n'apparaît que dans une seule contrainte, celle correspondant à l'équation (4.12). Comme la variable ξ sert à modéliser des inégalités strictes, nous en déduisons que la variable b_1^z est instanciée à la valeur 0 sauf si cette instanciation est incompatible avec la contrainte suivante :

$$\sum_{j=1}^{q} \omega_j^i y_j - \sum_{j=1}^{q} \omega_j^i z_j + M b_1^z > 0$$

Par conséquent, pour tout jeu de poids $\omega^i \in \Omega^i$ tel que $\sum_{j=1}^{q} \omega_j^i y_j - \sum_{j=1}^{q} \omega_j^i z_j > 0$, la variable b_1^z peut être librement instanciée à la valeur 0. En revanche, pour tous les autres jeux de poids $\omega^i \in \Omega^i$, la variable b_1^z doit être mise à 1 pour que cette dernière contrainte soit vérifiée (la constante positive M servant à compenser la valeur négative $\sum_{j=1}^{q} \omega_j^i y_j - \sum_{j=1}^{q} \omega_j^i z_j$). Ainsi, la variable b_1^z est égale à 1 si et seulement si nous avons $\sum_{j=1}^{q} \omega_j^i y_j \leq \sum_{j=1}^{q} \omega_j^i z_j$. Par ailleurs, par définition de Z_1, nous savons que $\sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j$ pour tout $\omega^i \in \Omega^i$. Finalement, nous venons de montrer que la variable b_1^z est instanciée à 1 si et seulement si l'alternative z satisfait l'équation (4.17).

De même, pour toute alternative $z \in Z_2$, nous savons que l'inégalité $\sum_{j=1}^{q} \omega_j^i z_j \geq \sum_{j=1}^{q} \omega_j^i y_j$ est vérifiée pour tout $\omega^i \in \Omega^i$ (par définition de Z_2). Par conséquent, pour montrer que la variable b_2^z est mise à 1 si et seulement si l'alternative z satisfait l'équation (4.17), il suffit de prouver que b_2^z est instanciée à 1 si et seulement si $\sum_{j=1}^{q} \omega_j^i x_j > \sum_{j=1}^{q} \omega_j^i z_j$. Remarquons que la variable b_2^z n'intervient que dans la contrainte suivante (cf. équation (4.13)) :

$$\sum_{j=1}^{q} \omega_j^i z_j - \sum_{j=1}^{q} \omega_j^i x_j + M b_2^z \geq 0$$

Ainsi, pour tous les jeux de poids $\omega^i \in \Omega^i$ tels que $\sum_{j=1}^q \omega_j^i z_j - \sum_{j=1}^q \omega_j^i x_j \geq 0$, la variable b_2^z peut être mise librement à la valeur 0 afin de minimiser la fonction objectif. Cependant, pour tous les autres jeux de poids $\omega^i \in \Omega^i$, la variable b_2^z est contrainte de valoir 1 pour que cette dernière contrainte soit vérifiée. Finalement, nous avons montré que la variable b_2^z vaut 1 si et seulement si $\sum_{j=1}^q \omega_j^i x_j > \sum_{j=1}^q \omega_j^i z_j$, i.e. si et seulement si l'alternative z vérifie l'équation (4.17).

Pour toute alternative $z \in Z_3$, observons que la variable b_3^z intervient uniquement dans la contrainte de l'équation (4.14). Cette contrainte impose b_3^z à valoir 1 si et seulement si les variables b_1^z et b_2^z sont toutes les deux égales à 1. Similairement au cas des alternatives de Z_1 (resp. Z_2), nous pouvons montrer que b_1^z (resp. b_2^z) est instanciée à 1 si et seulement si $\sum_{j=1}^q \omega_j^i y_j \leq \sum_{j=1}^q \omega_j^i z_j$ (resp. $\sum_{j=1}^q \omega_j^i x_j > \sum_{j=1}^q \omega_j^i z_j$), en utilisant la contrainte (4.12) (resp. (4.13)). Ainsi, l'équation (4.14) permet en réalité de modéliser l'opérateur logique de conjonction entre ces deux dernières conditions, ce qui nous permet de conclure que b_3^z vaut 1 si et seulement si l'alternative z vérifie l'équation (4.17).

En ce qui concerne l'alternative y, nous savons que l'inégalité $\sum_{j=1}^q \omega_j^i x_j \geq \sum_{j=1}^q \omega_j^i y_j$ est vraie pour tout $\omega^i \in \Omega^i$ puisque nous sommes dans le cas où $x \succsim_i^{Nec} y$. Ainsi, il s'agit d'instancier la variable b^y à la valeur 1 si et seulement si cette dernière inégalité est stricte. Ceci est en fait le rôle de la contrainte (4.11), qui fonctionne exactement comme celles de l'équation (4.13). □

Cette proposition nous offre la possibilité de calculer $\text{PMR}^i(x, y, \Omega^i)$ en utilisant la programmation linéaire en nombres entiers pour chaque agent $i \in N$ tel que $x \succsim_i^{Nec} y$. Intéressons nous à présent aux agents $i \in N$ pour lesquels $x \succsim_i^{Nec} y$ n'est pas vrai, si de tels agents existent.

• *Cas où $\neg(x \succsim_i^{Nec} y)$ (cf. figures 4.1b et 4.1c) :* nous avons vu précédemment que résoudre le problème d'optimisation $\text{PMR}^i(x, y, \Omega^i)$ dans le cas où $\neg(x \succ_i^{Nec} y) \wedge \neg(y \succ_i^{Nec} x)$ (i.e. figure 4.1c) peut se faire exactement de la même façon que pour le cas où $\neg(x \succ_i^{Nec} y) \wedge (y \succ_i^{Nec} x)$ (i.e. figure 4.1b). Par conséquent, nous nous intéressons ici uniquement au calcul de $\text{PMR}^i(x, y, \Omega^i)$ dans ce dernier cas. Rappelons alors que $\text{PMR}^i(x, y, \Omega^i) = \max_{\omega^i \in \Omega^i} |\{z \in \mathcal{X} : \sum_{j=1}^q \omega_j^i y_j > \sum_{j=1}^q \omega_j^i z_j \geq \sum_{j=1}^q \omega_j^i x_j\}|$. Il s'agit donc de déterminer un jeu de poids $\omega^i \in \Omega^i$ qui permet de maximiser le nombre d'alternatives $z \in \mathcal{X}$ telles que :

$$\sum_{j=1}^q \omega_j^i y_j > \sum_{j=1}^q \omega_j^i z_j \geq \sum_{j=1}^q \omega_j^i x_j$$

De façon similaire au cas où $x \succsim_i^{Nec} y$, nous observons ici que, pour tout $\omega^i \in \Omega^i$, aucune alternative z dans $P^y \cup B^x$ ne peut satisfaire les inégalités $\sum_{j=1}^q \omega_j^i y_j > \sum_{j=1}^q \omega_j^i z_j \geq \sum_{j=1}^q \omega_j^i x_j$, alors que toutes les alternatives $z \in M^{y,x}$ vérifient ces dernières inégalités. De cette simple observation, nous déduisons le résultat suivant :

$$\text{PMR}^i(x, y, \Omega^i) = |M^{y,x}| + \max_{\omega^i \in \Omega^i} |\{z \in Z_1 \cup Z_2 \cup Z_3 \cup \{x\} : \sum_{j=1}^q \omega_j^i y_j > \sum_{j=1}^q \omega_j^i z_j \geq \sum_{j=1}^q \omega_j^i x_j\}|$$

Pour obtenir la valeur $\text{PMR}^i(x, y, \Omega^i)$, il suffit donc de résoudre le problème d'optimisation suivant : $\max_{\omega^i \in \Omega^i} |\{z \in Z_1 \cup Z_2 \cup Z_3 \cup \{x\} : \sum_{j=1}^q \omega_j^i y_j > \sum_{j=1}^q \omega_j^i z_j \geq \sum_{j=1}^q \omega_j^i x_j\}|$. Pour y parvenir, nous

introduisons le programme linéaire en nombres entiers suivant (nommé $\text{PLNE}_{y,x}$ ci-après) :

$$
\begin{aligned}
\max \quad & b^x + \sum_{z \in Z_1} b_1^z + \sum_{z \in Z_2} b_2^z + \sum_{z \in Z_3} b_3^z \\
\text{s.c.} \quad & \sum_{j=1}^{q} \omega_j^i = 1 \\
& \sum_{j=1}^{q} \omega_j^i a_j - \sum_{j=1}^{q} \omega_j^i b_j \geq 0, \, \forall (a,b) \in \mathcal{P}_{\geq}^i \\
& \sum_{j=1}^{q} \omega_j^i a_j - \sum_{j=1}^{q} \omega_j^i b_j \geq \xi, \, \forall (a,b) \in \mathcal{P}_{>}^i \\
& \sum_{j=1}^{q} \omega_j^i y_j - \sum_{j=1}^{q} \omega_j^i x_j + (1 - b^x) M \geq \xi && (4.18) \\
& \sum_{j=1}^{q} \omega_j^i z_j - \sum_{j=1}^{q} \omega_j^i x_j + (1 - b_1^z) M \geq 0, \, \forall z \in Z_1 \cup Z_3 && (4.19) \\
& \sum_{j=1}^{q} \omega_j^i y_j - \sum_{j=1}^{q} \omega_j^i z_j + (1 - b_2^z) M \geq \xi, \, \forall z \in Z_2 \cup Z_3 && (4.20) \\
& b_3^z \leq b_1^z, \, \forall z \in Z_3 && (4.21) \\
& b_3^z \leq b_2^z, \, \forall z \in Z_3 && (4.22) \\
& \omega_j^i \geq 0, \, \forall j \in \{1, \ldots, q\} && (4.23) \\
& b^x \in \{0,1\}; \, b_1^z \in \{0,1\}, \, \forall z \in Z_1 \cup Z_3 && (4.24) \\
& b_2^z \in \{0,1\}, \, \forall z \in Z_2 \cup Z_3 \\
& b_3^z \in \{0,1\}, \, \forall z \in Z_3
\end{aligned}
$$

Proposition 34. *Si* $\neg(x \succsim_i^{Nec} y)$*, alors* $\text{PMR}^i(x, y, \Omega^i) = |M^{y,x}| + OPT$*, où* OPT *est la valeur optimale de* $PLNE_{y,x}$*.*

La preuve de cette proposition est délibérément omise du fait de sa forte similitude avec celle de la proposition 33. Soulignons toutefois que, puisque la fonction objectif est cette fois-ci à maximiser, les variables binaires b^x, b_1^z (pour $z \in Z_1$), b_2^z (pour $z \in Z_2$) et b_3^z (pour $z \in Z_3$) sont instanciées à 1 sauf si les contraintes du programme linéaire les obligent à valoir 0. Ce comportement est modélisé à l'aide des équations (4.18-4.22).

Finalement, nous avons proposé deux programmes linéaires en nombres entiers pour déterminer de manière exacte la valeur $\text{PMR}^i(x, y, \Omega^i)$ associée à chaque agent $i \in N$. Néanmoins, dans les situations de vote où le nombre d'alternatives possibles est relativement grand, la résolution de ces programmes engendre des temps de calcul excessifs, puisque ces derniers impliquent jusqu'à trois variables booléennes par alternative du problème. C'est pourquoi nous nous intéresserons aussi à la relaxation continue de ces programmes, c'est-à-dire aux programmes linéaires obtenus en remplaçant toutes les variables booléennes

par des variables continues devant appartenir à l'intervalle unité. L'intérêt de ces relaxations linéaires est de se ramener à des problèmes d'optimisation pouvant être résolus en temps polynomial par programmation linéaire. Cependant, la valeur optimale de ces programmes linéaires ne correspond plus exactement à la valeur $\text{PMR}^i(x, y, \Omega^i)$ mais nous fournit à la place une borne supérieure de cette dernière. Ces bornes supérieures peuvent être utilisées pour dériver des bornes supérieures sur les valeurs $\text{PMR}(x, y, \Omega)$ pour finalement obtenir une borne supérieure sur la valeur $\text{mMR}(\mathcal{X}, \Omega)$. En procédant de la sorte, nous obtenons une garantie de performance qui est certes moins forte que la valeur $\text{mMR}(\mathcal{X}, \Omega)$ exacte mais beaucoup plus facile à calculer. Par ailleurs, dans la Section 4.1.4 dédiée aux tests numériques, nous verrons que cette borne supérieure est relativement proche de la valeur réelle en pratique.

4.1.3 Stratégies d'élicitation

En fonction des ensembles Ω^i, $i \in N$, à considérer, il est possible que la garantie de performance quantifiée par la valeur $\text{mMR}(\mathcal{X}, \Omega)$ ne soit pas suffisamment satisfaisante pour les votants. Dans ces situations, il est préférable de réduire l'imprécision sur les jeux de poids $\omega^i, i \in N$, modélisant les préférences de nos agents afin d'améliorer la qualité de la recommandation. Il s'agit de poser des questions aux différents agents, soigneusement choisies les unes après les autres, de manière à obtenir l'inégalité $\text{mMR}(\mathcal{X}, \Omega) \leq \delta$ en sollicitant les votants le moins possible, où δ est un seuil de tolérance jugé acceptable par ces derniers. Rappelons que, dans le cas où $\delta = 0$, l'alternative qui sera recommandée est par définition est gagnant nécessaire par la méthode de Borda. Pour produire une question informative à une itération donnée, nous proposons de nous concentrer sur une paire d'alternative $(x^*, y^*) \in \mathcal{X}^2$ telle que :

$$x^* \in \arg\min_{x \in \mathcal{X}} \text{MR}(x, \mathcal{X}, \Omega)$$
$$y^* \in \arg\max_{y \in \mathcal{X}} \text{PMR}(x^*, y, \Omega)$$

Poser une question permettant de diminuer significativement la valeur $\text{PMR}(x^*, y^*, \Omega)$ peut en effet réduire aussi nettement la valeur $\text{mMR}(\mathcal{X}, \Omega)$ courante, car nous avons $\text{mMR}(\mathcal{X}, \Omega) = \text{PMR}(x^*, y^*, \Omega)$ par définition. Rappelons que cette idée est au cœur de la stratégie d'élicitation CSS (présentée section 1.4.2) qui s'est révélée très efficace en pratique dans le cadre de la problématique de choix mono-agent. Cette dernière stratégie consiste à demander au décideur de comparer directement les alternatives x^* et y^*, la réponse permettant de faire passer soit $\text{PMR}(x^*, y^*, \Omega)$ soit $\text{PMR}(y^*, x^*, \Omega)$ en dessous de la valeur zéro. Dans notre contexte de décision collective avec la méthode de Borda, la valeur $\text{PMR}(x, y, \Omega)$, pour deux alternatives $x, y \in \mathcal{X}$ quelconques, ne dépend pas uniquement des préférences de nos agents concernant les alternatives x et y. En particulier, nous avons vu que la donnée $x \succsim_i^{Nec} y$ pour un agent $i \in N$ donné ne suffit pas nécessairement à connaître la contribution $\text{PMR}^i(x, y, \Omega)$ de cet agent. Cette dernière valeur dépend en réalité de la position relative de toutes les alternatives $z \in \mathcal{X}$ par rapport aux alternatives x et y, dans le classement induit par la relation de préférence $\succsim_i$ de l'agent i. Nous proposons ci-après deux stratégies d'élicitation de complexité différente, visant à réduire la valeur $\text{PMR}(x^*_\Omega, y^*_\Omega, \Omega)$.

Multicritère-CSS0 (M-CSS0). Cette stratégie vise à choisir une paire (agent, question) telle que la question permette de réduire la contribution de l'agent à la valeur $\text{PMR}(x^*_\Omega, y^*_\Omega, \Omega)$. Plus précisément, nous commençons par sélectionner un agent $i \in N$ au hasard puis nous cherchons une question permettant de diminuer la valeur $\text{PMR}^i(x^*_\Omega, y^*_\Omega, \Omega)$ (s'il en existe) en procédant comme décrit ci-dessous :

1) Cas où $x^* \succsim_i^{Nec} y^*$ (cf. figure 4.1a) : rappelons que, dans ce cas, nous avons $\text{PMR}^i(x^*, y^*, \Omega^i) = -|M^{x^*,y^*}| - \min_{\omega^i \in \Omega^i} |\{z \in Z_1 \cup Z_2 \cup Z_3 \cup \{y^*\} : \sum_{j=1}^q \omega^i_j x^*_j > \sum_{j=1}^q \omega^i_j z_j \geq \sum_{j=1}^q \omega^i_j y^*_j\}|$. Nous distinguons alors les deux scénarios suivants :

- *Cas où $Z_1 \cup Z_2 \cup Z_3 = \emptyset$:* si nous avons $\neg(x^* \succ_i^{Nec} y^*)$, alors nous demandons à l'agent i s'il préfère strictement l'alternative x^* à l'alternative y^*. Si en revanche $x^* \succ_i^{Nec} y^*$ est vraie, alors la différence de scores entre l'alternative x^* et y^* pour l'agent i vaut exactement $-|M^{x^*,y^*}| - 1$. Dans ce cas, la contribution de l'agent i à la valeur $\text{PMR}(x^*, y^*, \Omega)$ ne peut pas diminuer, et donc il est préférable de ne pas lui poser de question à cette étape. La stratégie M-CSS0 sélectionne alors un autre agent au hasard pour l'interroger.
- *Cas où $Z_1 \cup Z_2 \cup Z_3 \neq \emptyset$* : une alternative z^* dans $Z_1 \cup Z_2 \cup Z_3$ est choisie au hasard par la procédure. Par définition de ces ensembles, notre connaissance actuelle sur les préférences de l'agent i ne nous permet pas de conclure si $x^*_j \succ_i z^* \succsim_i y^*$ est vraie ou non. Plus précisément, si $z^* \in Z_1$, alors l'inégalité $\sum_{j=1}^q \omega^i_j x^*_j > \sum_{j=1}^q \omega^i_j z^*_j$ est vérifiée pour tout $\omega^i \in \Omega^i$ mais il existe au moins un jeu de poids $\omega^i \in \Omega^i$ tel que $\sum_{j=1}^q \omega^i_j z^*_j < \sum_{j=1}^q \omega^i_j y^*_j$. Par conséquent, si $z^* \in Z_1$, alors M-CSS0 demande à l'agent i s'il préfère (faiblement) z^* à y^* pour pouvoir obtenir l'information manquante. Similairement, si $z^* \in Z_2$, alors l'inégalité $\sum_{j=1}^q \omega^i_j z^*_j \geq \sum_{j=1}^q \omega^i_j y^*_j$ est vraie pour tout $\omega^i \in \Omega^i$ mais nous avons $\sum_{j=1}^q \omega^i_j z^*_j \geq \sum_{j=1}^q \omega^i_j x^*_j$ pour au moins un jeu de poids $\omega^i \in \Omega^i$. De ce fait, si $z^* \in Z_2$, alors la procédure demande à l'agent i s'il préfère strictement x^* à z^*. Dans le cas où $z^* \in Z_3$, aucune des deux inégalités n'est vraie pour tous les jeux de poids $\omega^i \in \Omega^i$ (sinon z^* appartiendrait à Z_1 ou à Z_2). Par conséquent, nous avons ici le choix entre deux questions, demander à l'agent i s'il préfère (faiblement) z^* à y^* ou lui demander s'il préfère strictement x^* à z^* (le choix entre ces deux questions est ici laissé au hasard).

2) Cas où $\neg(x^* \succsim_i^{Nec} y^*) \wedge (y^* \succsim_i^{Nec} x^*)$ (cf. figures 4.1b) : commençons par rappeler que dans ce cas $\text{PMR}^i(x^*, y^*, \Omega^i) = |M^{y^*,x^*}| + \max_{\omega^i \in \Omega^i} |\{z \in Z_1 \cup Z_2 \cup Z_3 \cup \{x\} : \sum_{j=1}^q \omega^i_j y^*_j > \sum_{j=1}^q \omega^i_j z_j \geq \sum_{j=1}^q \omega^i_j x^*_j\}|$. Nous distinguons ici aussi les deux scénarios suivants :

- *Cas où $Z_1 \cup Z_2 \cup Z_3 = \emptyset$* : si $\neg(y^* \succ_i^{Nec} x^*)$, alors nous demandons à l'agent i s'il préfère strictement y^* à x^*. Par contre, si $y^* \succ_i^{Nec} x^*$ à la place, alors la différence de scores entre x^* et y^* pour cet agent est forcément égale à $|M^{y^*,x^*}| + 1$. Dans ce cas, M-CSS0 choisit un autre agent au hasard.
- *Cas où $Z_1 \cup Z_2 \cup Z_3 \neq \emptyset$* : une alternative z^* dans $Z_1 \cup Z_2 \cup Z_3$ est choisie au hasard et nous cherchons à savoir si $y^* \succ_i z^* \succsim_i x^*$ est vraie ou non. Plus précisément, si $z^* \in Z_1$, alors nous demandons à l'agent i s'il préfère (faiblement) z^* à x^*. Si $z^* \in Z_2$, alors nous lui demandons si y^* est strictement meilleure que z^*. Enfin, si $z^* \in Z_3$, alors nous choisissons au hasard une des deux questions précédentes.

3) Cas où $\neg(x^* \succsim_i^{Nec} y^*) \wedge \neg(y^* \succsim_i^{Nec} x^*)$ *(cf. figure 4.1c)* : dans ce cas, les alternatives x^* et y^* sont incomparables pour le système, et donc nous demandons à l'agent i de comparer directement ces deux alternatives pour pouvoir se ramener aux cas précédents.

Multicritère-CSS1 (M-CSS1). Cette stratégie est une adaptation de l'heuristique proposée par Lu and Boutilier [2011b] à notre contexte de décision collective avec évaluation multicritère. L'objectif est de trouver la question avec le plus grand potentiel de réduction de la valeur $\text{PMR}(x^*, y^*, \Omega)$. Plus précisément, au lieu de sélectionner l'agent i ainsi que l'alternative $z^* \in Z_1 \cup Z_2 \cup Z_3$ au hasard (comme dans la stratégie M-CSS0), il s'agit ici d'envisager toutes les paires (agent, question) possibles pour pouvoir déterminer celle qui offre la meilleure réduction de la valeur $\text{PMR}(x^*, y^*, \Omega)$ dans le scénario de réponse le plus favorable. Par conséquent, cette stratégie nécessite de calculer les regrets résultants dans tous les scénarios de réponse possibles et ce pour chaque paire (agent, question) envisageable ; cette méthode incrémentale est donc plus gourmande que la stratégie M-CSS0.

4.1.4 Résultats expérimentaux

Dans cette sous-section, nous évaluons la pertinence des méthodes interactives M-CSS0 et M-CSS1 (cf. Section 4.1.3) que nous avons proposées pour la détermination du vainqueur par la méthode de Borda dans un contexte de vote où les préférences des agents sur les alternatives multicritères sont imprécisément connues. Dans nos expériences, les vecteurs de performances associés aux alternatives du problème ont été engendrés de manière aléatoire dans $[0, 1]^q$. Chacune de nos simulations commence avec aucune information disponible sur les préférences des agents. À chaque itération, une question est posée à un agent $i \in N$ particulier et la réponse de celui-ci est simulée à l'aide du jeu de poids ω^i que nous avons engendré aléatoirement en début de simulation. Les tests ont été réalisés sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire et les optimisations de regrets ont été effectuées par le solveur Gurobi depuis un programme écrit en Java.

Nos premières expériences ont pour objectif d'estimer l'apport d'une représentation compacte des préférences sur domaine multicritère. C'est pourquoi nous avons choisi de comparer ici nos stratégies M-CSS0 et M-CSS1 à la procédure d'élicitation incrémentale proposée dans Lu and Boutilier [2011b] (nommée CSS1 ci-après), qui ne fait aucune hypothèse sur la structure des préférences de nos agents [2]. Dans la figure 4.2, nous reportons les valeurs $\text{mMR}(\mathcal{X}, \Omega)$ observées aux différentes itérations de ces procédures d'élicitation incrémentale. Dans cette figure, les valeurs de regrets sont exprimées sur une échelle normalisée, la valeur 1 correspondant à la valeur mMR initiale (c'est-à-dire celle obtenue sans aucune information sur les préférences des agents). Rappelons que lorsque la valeur mMR vaut 0, la procédure a en réalité identifié un vainqueur nécessaire au sens de Borda.

2. Ainsi, les stratégies CSS1 et M-CSS1 adoptent exactement la même stratégie de sélection de questions mais, contrairement à CSS1 qui utilise uniquement les données de préférence explicitement formulées par les agents, la stratégie M-CSS1 exploite la structure multicritère du problème pour identifier les ensembles Z_1, Z_2, etc., et en déduire les valeurs de regrets.

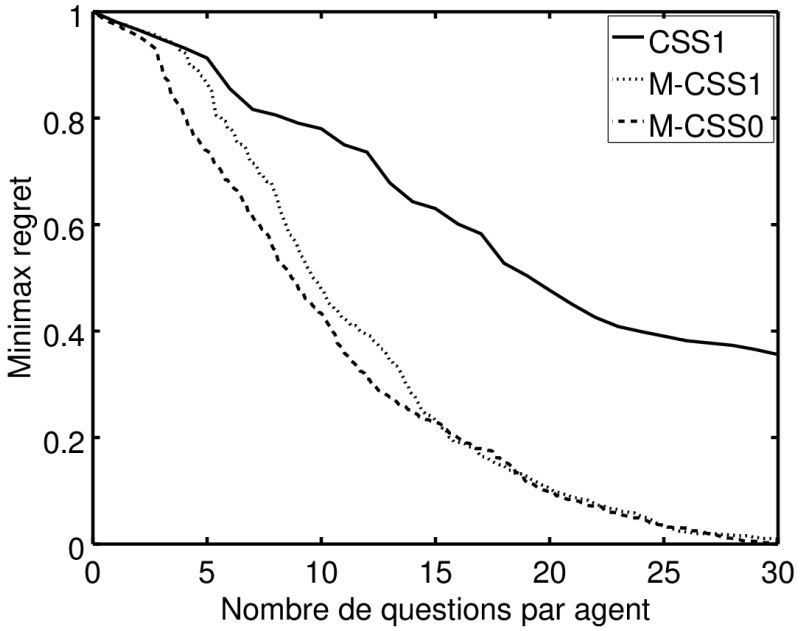


FIGURE 4.2 – Comparaison des stratégies d'élicitation (30 alternatives, 5 critères, 10 agents, 30 tests).

Dans la figure 4.2, nous observons que la valeur mMR diminue plus lentement avec la stratégie CSS1 qu'avec sa version multicritère M-CSS1. Par exemple, après 20 questions par agent en moyenne, la valeur mMR est toujours au dessus de 40% du regret initial avec la stratégie CSS1 alors que cette valeur est en dessous des 10% avec la stratégie M-CSS1. Par ailleurs, après environ 30 questions par agent, la valeur mMR est toujours aux alentours des 40% avec la stratégie CSS1 alors que la stratégie M-CSS1 est capable de déterminer un vainqueur nécessaire à ce moment là. Enfin, nous constatons assez étonnamment que l'heuristique utilisée par M-CSS1 n'est finalement pas plus efficace que la stratégie M-CSS0 qui comporte une partie aléatoire. Par conséquent, nous utiliserons exclusivement la stratégie M-CSS0 dans nos expériences suivantes puisque M-CSS1 requiert bien plus de calculs que M-CSS0 sans se révéler plus informative.

Dans la Section 4.1.2, nous avons montré comment les valeurs $\text{mMR}(\mathcal{X}, \Omega)$ pouvaient être calculées de manière exacte en faisant usage de la programmation linéaire en nombres entiers (cf. propositions 33 et 34). Rappelons que la relaxation linéaire des programmes en nombre entiers correspondants permet de déduire des bornes supérieures des valeurs $\text{mMR}(\mathcal{X}, \Omega)$ en des temps plus courts. C'est pourquoi les expériences suivantes ont pour objectif de quantifier la perte de précision provoquée par l'utilisation de la relaxation linéaire de ces programmes durant l'élicitation des préférences. Dans la figure 4.3, nous reportons à chaque itération la valeur $\text{mMR}(\mathcal{X}, \Omega)$, la borne supérieure de cette valeur obtenue par relaxation linéaire, et le regret réel en terme de scores de Borda ; Les regrets sont ici aussi normalisés, la valeur 1 correspondant à la borne supérieure initiale de la valeur mMR.

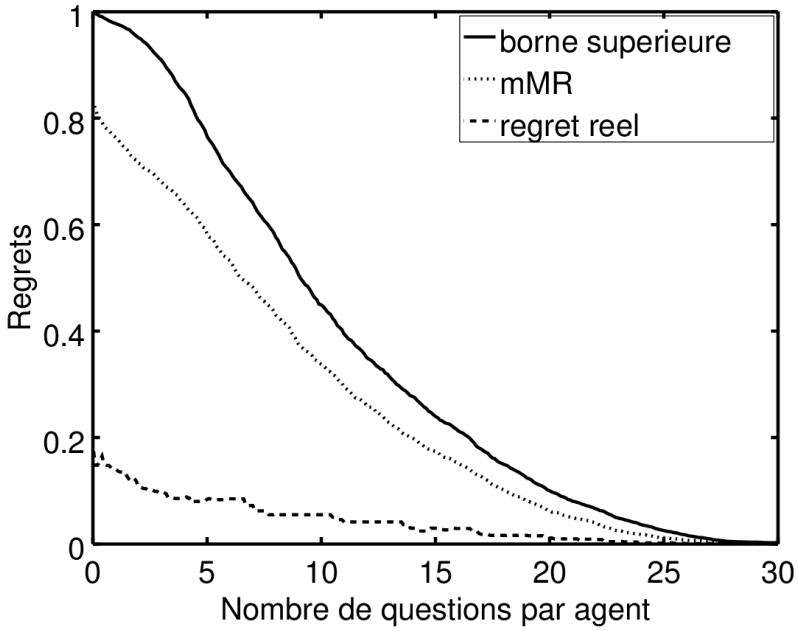
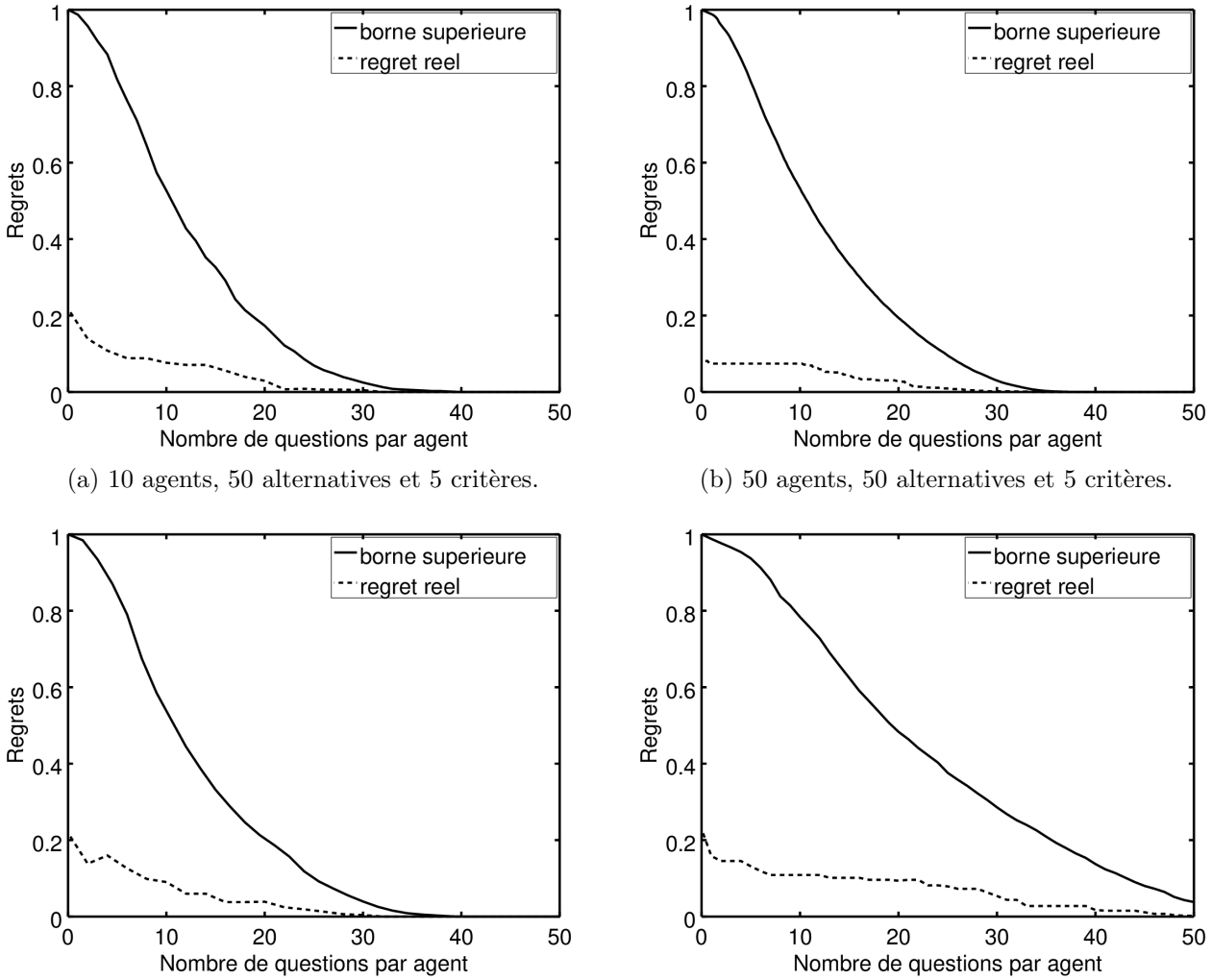


FIGURE 4.3 – Comparaison entre la valeur mMR, sa borne supérieure obtenue par relaxation linéaire et le regret réel, observés à chaque itération de la stratégie de génération de questions M-CSS0 (30 alternatives, 5 critères, 10 agents, 30 tests).

Dans la figure 4.3, nous observons que les relaxations linéaires fournissent en pratique une borne supérieure relativement proche de la valeur exacte. Cette borne devient par ailleurs de plus en plus précise au fur et à mesure des interactions. De ce fait, il semble que travailler avec cette borne supérieure ne nous fait pas perdre en efficacité d'élicitation. Signalons aussi que la version relaxée permet de réduire significativement les temps de calculs en pratique. À titre d'exemple, sans aucune information sur les préférences des agents, la borne supérieure de la valeur mMR est obtenue en seulement 1 seconde contre 30 secondes en moyenne pour l'obtention de la valeur exacte, dans des problèmes impliquant 30 alternatives, 5 critères et 10 agents. Enfin, nous voyons sur cette même figure que le regret réel (ici associé à la recommandation de l'alternative minimisant la valeur MR approchée) est en pratique relativement faible en comparaison avec la borne supérieure fournie par le mMR.

Les expériences suivantes ont pour objectif d'évaluer le nombre de questions produites par la stratégie M-CSS0 (avec mMR approchées) sur de plus grandes instances. Plus précisément, nous allons faire varier le nombre d'alternatives, le nombre d'agents ainsi que le nombre de critères du problème. Les résultats que nous avons obtenus sont reportés dans la figure 4.4. Dans cette figure, nous voyons que, dans tous les situations considérées, la stratégie M-CSS0 est capable de déterminer un gagnant nécessaire au sens de Borda après un nombre relativement raisonnable de questions par agent. Notons toutefois que le nombre de critères semble être la donnée ayant le plus grand impact sur le nombre de questions engendrées. Ce constant s'explique par le fait que ce paramètre contrôle directement le nombre de poids ω_j^i à apprendre.



(a) 10 agents, 50 alternatives et 5 critères.

(b) 50 agents, 50 alternatives et 5 critères.

(c) 10 agents, 100 alternatives et 5 critères.

(d) 10 agents, 50 alternatives et 7 critères.

FIGURE 4.4 – Performances de la stratégie M-CSS0 appliquée à la borne supérieure du mMR obtenue par relaxation linéaire (30 tests).

4.1.5 Rangement par élicitation incrémentale

Dans de nombreuses situations, il est utile de connaître les k meilleures alternatives du problème, par exemple lorsque plusieurs postes sont à pourvoir au sein d'une entreprise. Quand le profil de préférences est parfaitement connu, les alternatives peuvent être rangées de la meilleure à la plus mauvaise en suivant les scores attribués par la méthode de vote scorage (ici la méthode de Borda). Cependant, dans les situations où les préférences individuelles ne sont pas connues de manière précise, les informations préférentielles disponibles peuvent être insuffisantes pour déduire le rangement représentant les préférences collectives. Dans ces situations, il s'agit de poser des questions aux agents permettant de détecter le plus rapidement possible le rangement des alternatives associé à la méthode de vote considérée.

Comme le nombre de rangements possibles est combinatoire, nous proposons de déterminer les k meilleures alternatives du problème en utilisant une méthode par choix itérés (comme dans la Section 2.3.1). Plus précisément, dans notre contexte de vote avec la méthode de Borda, nous commençons par engendrer des questions selon la stratégie M-CSS0 tant que la borne supérieure de la valeur mMR est strictement plus grande que zéro. Dès que ce n'est plus le cas, la procédure a en fait détecté la meilleure alternative au sens de Borda et donc celle-ci peut être placée à la première position du rangement final. Cette alternative est ensuite retirée de l'ensemble des alternatives [3] afin de pouvoir itérer le processus et détecter la seconde meilleure alternative, etc.

Nous présentons maintenant quelques résultats expérimentaux pour évaluer les performances cette méthode de rangement incrémental par choix itérés. Dans la figure 4.5, nous donnons le nombre moyen de questions produites par cette méthode pour déterminer les k meilleures alternatives du problème, avec k variant de 0 à 10. Nous observons que le coût marginal pour identifier l'alternative en position k dans le rangement final diminue lorsque la position k augmente ; autrement dit, il nous faut de moins en moins d'informations préférentielles pour trouver la prochaine alternative à insérer dans le rangement final. Par ailleurs, nous voyons que la majeure partie des questions est engendrée à la première itération, c'est-à-dire lors de la détermination de la meilleure alternative ; ceci illustre l'intérêt de l'approche incrémentale pour résoudre des problèmes de rangement.

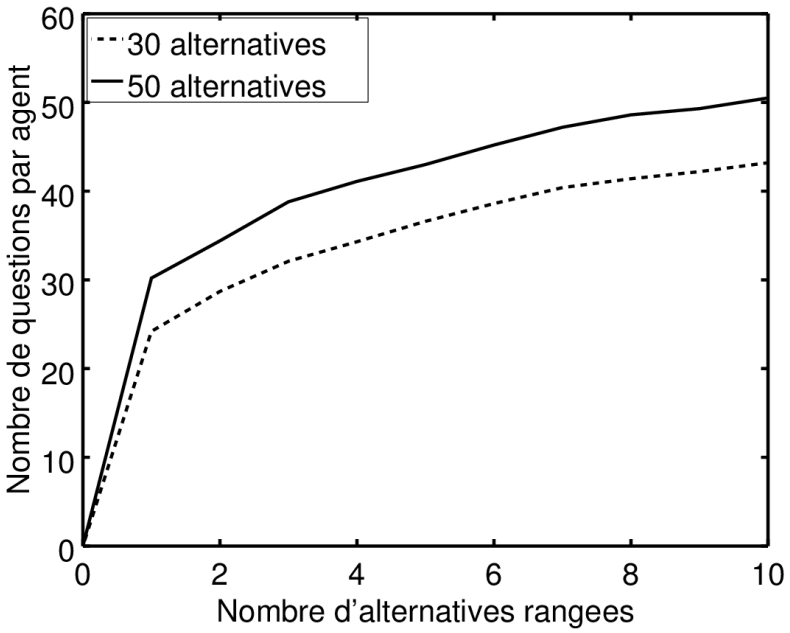


FIGURE 4.5 – Performance de la méthode de rangement incrémental par choix itérés fondés sur la stratégie M-CSS0 et la borne supérieure du mMR obtenue par relaxation linéaire (20 agents, 5 critères, 30 tests).

3. Cette alternative reste toutefois associée à des variables binaires de types b_1^z, b_2^z, b_3^z, dans les programmes linéaires en nombres entiers que nous utilisons pour calculer les valeurs de PMR (car le score de Borda d'une alternative dépend de toutes les autres alternatives du problème).

À présent, nous souhaitons comparer les performances de notre algorithme de rangement incrémental (nommé RI ci-après) avec celles d'autres méthodes de rangement. Par exemple, pour déterminer le rangement induit par les scores de Borda, nous pouvons envisager une méthode en deux phases consistant tout d'abord à éliciter le profil de préférences complet puis à calculer les scores de Borda associés au profil de préférences ainsi obtenu. Cette première phase peut être réalisée de différentes manières. Nous considérons ici les deux méthodes suivantes :

- **S0** : pour chaque agent, nous déterminons le rangement des alternatives selon ses préférences individuelles en utilisant un algorithme de tri standard : une question lui est posée par comparaison effectuée durant le déroulement de l'algorithme de tri. Cette méthode nécessite donc $O(|\mathcal{X}| \log_2(|\mathcal{X}|))$ questions par agent.
- **S1** : les rangements individuels sont ici déterminés par élicitation incrémentale. Plus précisément, nous utilisons l'approche incrémentale par choix itérés dédiée au cas mono-agent et fondée sur la stratégie CSS (présentée en 1.2.4).

Dans la table 4.1, nous donnons le nombre moyen de questions par agent qui a été nécessaire pour identifier les 10 meilleures alternatives. Dans cette table, nous faisons varier n le nombre d'agents, q le nombre de critères ainsi que $|\mathcal{X}|$ le nombre d'alternatives du problème. Nous constatons que notre algorithme de rangement incrémental fondé sur les scores de Borda est plus performant que les méthodes en deux phases S0 et S1 sur toutes les instances considérées. Notons toutefois que le nombre de questions posées augmente drastiquement avec le nombre de critères du problème.

n	$\lvert\mathcal{X}\rvert$	q	RI	S0	S1
10	30	5	43.3	147.2	58.7
10	50	5	43.7	282.2	67.4
100	30	5	51.1	147.2	87.2
10	30	10	93.3	147.2	178.2

TABLE 4.1 – Comparaison des méthodes de rangement sur la base du nombre de questions (30 tests).

Dans cette section, nous nous sommes intéressés à des situations de vote où les préférences des agents sont modélisées à l'aide d'une somme pondérée dont les poids devaient être précisés. Dans ce contexte, nous avons proposé des algorithmes interactifs permettant de déterminer de manière efficace un gagnant nécessaire pour la méthode Borda ; nous avons aussi proposé une méthode par choix itérés fondée sur les scores de Borda pour la problématique de rangement. Bien que ces méthodes se soient révélées relativement efficaces en pratique, celles-ci deviennent impraticables dans des contextes de vote combinatoire, dans lesquels l'ensemble des alternatives possibles est défini implicitement tout en étant de très grande taille. En effet, dans ces contextes, il n'est plus envisageable d'énumérer toutes les paires d'alternatives $(x, y) \in \mathcal{X}^2$ pour calculer la valeur $\mathrm{mMR}(\mathcal{X}, \Omega)$ à partir des valeurs $\mathrm{PMR}(x, y, \Omega)$. C'est pourquoi nous nous attaquons à la problématique de vote sur domaine combinatoire en présence de préférences incomplètes dans la sous-section suivante.

4.2 Un problème sur domaine combinatoire : sac à dos multi-agents et vote par approbation

La décision collective sur domaine combinatoire intervient dans de nombreux contextes, comme dans le cadre de la planification d'investissement, de l'allocation de ressources ou encore de la configuration par groupe. Cette tâche est généralement très complexe car il s'agit d'identifier, parmi un nombre combinatoire de possibilités, la meilleure alternative du point de vue des différents agents intervenant dans la prise de décision (chacun d'eux ayant sa propre rationalité et son propre système de valeurs). En raison des aspects stratégiques souvent liés à la prise de décision collective et des divergences possibles entre les valeurs individuelles, il paraît essentiel de développer des méthodes et des outils formels permettant de modéliser les préférences des agents sur domaine combinatoire et de résoudre efficacement les problèmes de décision associés. Cette observation a motivé ces dernières années de nombreux travaux dans le domaine du choix social computationnel (e.g., [Brandt et al., 2016] chapitre 9). Dans cette section, nous allons étudier le problème du sac à dos multi-agents qui, étant donné un ensemble fini d'objets de coût positif, consiste à déterminer le sous-ensemble d'objets maximisant l'utilité des agents sous une contrainte de budget donnée. Ce problème est relativement standard en optimisation combinatoire avec de nombreuses applications, comme la sélection de projets, la gestion de portefeuille ou encore l'élection de comité (e.g., [Lu and Boutilier, 2011c, Elkind et al., 2014, Oren and Lucier, 2014, Skowron et al., 2015a]).

Dans le cadre de problèmes d'optimisation combinatoire, il n'est pas raisonnable d'espérer que les agents nous fournissent leurs préférences complètes sur l'espace des solutions. Pour pouvoir résoudre efficacement ces problèmes, il est nécessaire de gérer les préférences individuelles et collectives à l'aide de représentations compactes. En l'occurrence, pour le problème du sac à dos mono-agent, les préférences sur les sous-ensembles d'objets possibles sont traditionnellement modélisées en utilisant une fonction d'utilité additive. Plus précisément, l'utilité d'un sous-ensemble d'objets pour le décideur est définie comme la somme des utilités de chaque élément de ce sous-ensemble. Dans un contexte multi-agents, même si des représentations numériques des préférences sont disponibles sous la forme de fonctions d'utilité, celles-ci sont généralement construites de manière indépendante pour chaque agent, ce qui ne permet pas de comparer les satisfactions des individus de manière intelligible. En particulier, définir l'utilité collective comme la somme des utilités des agents (critère utilitaire) n'aurait aucun sens dans ce contexte, et ce même en supposant que les utilités sont exprimées sur la même échelle. Pour nous en convaincre, intéressons-nous à l'exemple suivant :

Exemple 29. *Considérons un problème de sac à dos multi-agents impliquant* 3 *objets et* 2 *agents. Tout sous-ensemble d'objets possible peut être caractérisé par un vecteur* $x = (x_1, x_2, x_3) \in \{0,1\}^3$ *avec* $x_j = 1$ *si et seulement si le* $j^{ème}$ *objet est dans le sous-ensemble en question. Les* 2 *agents doivent se mettre d'accord sur un sous-ensemble d'objets* x *vérifiant la contrainte de budget suivante :* $x_1 + x_2 + x_3 \leq 2$. *Un tel problème pourrait se poser par exemple pour la formation d'un comité de taille* 2*, étant donnés* 3 *candidats et* 2 *votants. Supposons que les fonctions d'utilité de nos deux agents sont respectivement*

définies par $u^1(x) = u^1_1x_1 + u^1_2x_2 + u^1_3x_3$ *et* $u^2(x) = u^2_1x_1 + u^2_2x_2 + u^2_3x_3$*, où* u^i_j *représente l'utilité de l'objet* j *pour l'agent* $i \in \{1,2\}$*. Supposons par ailleurs que les préférences individuelles* $\succ_i$*,* $i \in \{1,2\}$*, sur les sous-ensembles possibles sont les suivantes :*

$$\{2,3\} \succ_1 \{1,3\} \succ_1 \{1,2\}$$
$$\{1,2\} \succ_2 \{1,3\} \succ_2 \{2,3\}$$

Une représentation numérique possible de ces préférences, utilisant la même échelle, est la suivante : $(u^1_1, u^1_2, u^1_3) = (1,2,4)$ *et* $(u^2_1, u^2_2, u^2_3) = (4,2,1)$*. Cette représentation numérique attribue les utilités suivantes aux sous-ensembles de taille* 2 *:*

	$\{1,2\}$	$\{2,3\}$	$\{1,3\}$
u^1	3	6	5
u^2	6	3	5

Observons que ces valeurs sont bien cohérentes avec les relations de préférence $\succ_1$ *et* $\succ_2$*. Avec le critère utilitaire, nous déduisons de ce tableau que la solution optimale est le sous-ensemble* $\{1,3\}$ *car celui-ci maximise la somme des utilités* $(5+5 = 10 > 3+6 = 9)$*. Changeons à présent la représentation numérique en remplaçant les valeurs* $(1,2,4)$ *par les valeurs* $(0,3,4)$*. Dans ce cas, nous obtenons deux nouvelles fonctions d'utilité caractérisées par* $(u^1_1, u^1_2, u^1_3) = (0,3,4)$ *et* $(u^2_1, u^2_2, u^2_3) = (4,3,0)$*, ce qui nous donne les utilités suivantes :*

	$\{1,2\}$	$\{2,3\}$	$\{1,3\}$
u^1	3	7	4
u^2	7	3	4

Soulignons que cette représentation numérique est elle aussi compatible avec les relations de préférence $\succ_1$ *et* $\succ_2$*. Avec cette représentation numérique, nous devons maintenant admettre que le sous-ensemble* $\{1,3\}$ *constitue le plus mauvais choix possible, ce dernier minimisant la somme des utilités* $(4+4 = 8 < 7+3 = 10)$*. De ce fait, nous constatons que le résultat dépend ici fortement de la représentation numérique choisie par le système : la décision finale semble plus être la conséquence d'un choix arbitraire de la représentation numérique plutôt qu'une conclusion robuste induite par le profil de préférences* $(\succ_1, \succ_2)$ *observé. Notons que le problème persiste même après avoir normalisé les utilités* u^1 *et* u^2 *de sorte que la valeur* 1 *soit attribuée à l'ensemble* $\{1,2,3\}$*. Par ailleurs, il est important de souligner que ce problème ne se pose pas uniquement dans le cadre d'une agrégation additive des préférences ; des exemples similaires peuvent en effet être construits pour d'autres agrégateurs, comme par exemple le minimum (critère égalitaire).*

Lorsque les échelles d'utilité individuelles ne sont pas commensurables et/ou pas suffisamment riches pour pouvoir construire une fonction d'utilité sociale, il semble naturel de recourir à l'utilisation d'une règle de vote car celles-ci réalisent une agrégation ordinale des préférences. Pour mettre en œuvre une procédure de vote, il est en effet inutile de savoir comment comparer les bien-être individuels, il suffit

de connaître les relations de préférence individuelle sur les solutions du problème. À ce niveau, il est important de souligner que mettre en place une procédure de vote dans notre contexte n'entre pas en contradiction avec la représentation des valeurs individuelles par des utilités. En effet, les fonctions d'utilité individuelles vont ici être vues comme des représentations commodes et compactes des relations de préférence individuelle, et leur utilisation permettra de réduire significativement le nombre de questions à poser aux agents pour déterminer le gagnant de l'élection. Notons que certaines contributions récentes envisagent l'utilisation des règles de vote à des fins différentes, étudiant par exemple leur capacité à approximer le gagnant d'une élection au sens du critère utilitaire (e.g., [Procaccia and Rosenschein, 2006b, Caragiannis and Procaccia, 2011, Boutilier et al., 2015]).

Quand les préférences individuelles ne sont pas totalement connues, nous pouvons nous intéresser à la problématique de détermination des gagnants possibles et nécessaires à partir d'un profil de préférences partiel (e.g., [Konczak and Lang, 2005, Xia and Conitzer, 2011, Lang et al., 2012]). Dans ce contexte, nous pouvons aussi chercher à identifier le gagnant de l'élection par élicitation incrémentale pour y parvenir avec un nombre réduit d'interactions avec les agents (e.g., [Kalech et al., 2010, Lu and Boutilier, 2011b, Ding and Lin, 2013, Dery et al., 2014, 2016]). Toutefois, dans le problème du sac à dos multi-agents, les solutions possibles sont en très grand nombre et définies de manière implicite, ce qui constitue un défi supplémentaire pour la détermination des gagnants. Notre objectif ici est de proposer une procédure de vote incrémentale dans laquelle les préférences individuelles sont révélées progressivement jusqu'à pouvoir déterminer le vainqueur de l'élection. Cette procédure doit tirer parti du fait que les préférences individuelles sont représentables par des fonctions d'utilité additives pour pouvoir déterminer rapidement la bonne décision. En effet, la linéarité des préférences permet de caractériser l'ensemble des relations de préférence compatibles avec un ordre partiel donné par un polyèdre convexe dans l'espace des utilités. Ceci rend possible le recours à la programmation mathématique pour explorer toutes les complétions possibles d'un profil de préférences partiel quelconque.

Dans cette section, nous nous concentrons sur le vote par approbation qui est une règle de vote suffisamment simple pour être envisagée sur domaine combinatoire et qui peut être décisive même lorsque le profil de préférences est partiellement connu. Rappelons que le vote par approbation permet aux agents de voter pour autant de solutions qu'ils le souhaitent, la solution gagnante étant celle qui reçoit le plus de votes en sa faveur (cf. section 1.2.3 pour une description détaillée). Pour le problème du sac à dos multi-agents, il suffit donc d'identifier tous les sous-ensembles d'objets que chaque agent approuve pour pouvoir déterminer le sous-ensemble d'objets gagnant. Soulignons que cette problématique diffère du vote par approbation pour l'élection de gagnants multiples, qui consiste à demander aux agents de donner leur approbation à des objets pris séparément mais jamais de s'exprimer sur des sous-ensembles d'objets (e.g., [Kilgour, 2010, Aziz et al., 2015a,c]). Plus généralement, la plupart des approches proposées pour le vote à gagnants multiples suppose que les préférences individuelles sur les objets suffisent à expliquer les préférences sur les sous-ensembles d'objets, parce que les agents tirent satisfaction uniquement de leur objet préféré dans le sous-ensemble (e.g., [Chamberlin and Courant, 1983, Monroe, 1995, Potthoff and Brams, 1998, Procaccia et al., 2008, Meir et al., 2008, Lu and Boutilier,

2011c, Betzler et al., 2013, Elkind et al., 2014, Skowron et al., 2015b], et [Lu and Boutilier, 2013] pour l'élicitation incrémentale des préférences). Cette hypothèse semble plutôt bien adaptée aux problèmes d'élection de représentants, les objets du problème correspondant alors aux différents candidats qui se présentent. Cependant, même pour ces problèmes, cette hypothèse peut paraître assez restrictive dans certaines situations. Par exemple, il est impossible de modéliser le comportement suivant : la présence du candidat préféré parmi les élus ne permet pas de contrebalancer le fait que plusieurs candidats détestés en fassent aussi partie. Ce dernier fait est illustré dans l'exemple suivant :

Exemple 30. *Considérons un problème de décision collective où* 3 *votants doivent choisir* 3 *candidats parmi les* 5 *en lice pour former un comité de représentants. Supposons que les préférences de nos votants sur les différents candidats sont les suivantes :*

$$1 \succ_1 2 \succ_1 3 \succ_1 4 \succ_1 5$$
$$4 \succ_2 2 \succ_2 3 \succ_2 5 \succ_2 1$$
$$5 \succ_3 2 \succ_3 3 \succ_3 1 \succ_3 4$$

Nous allons montrer que la célèbre règle de Chamberlin-Courant (e.g., [Chamberlin and Courant, 1983]), conçue pour les élections à gagnants multiples, peut conduire ici à l'élection d'un comité approuvé par aucun votant. Pour ce faire, commençons par décrire cette règle de vote. L'idée principale est de fournir une répartition optimale des votants sur les membres du comité en utilisant une fonction de satisfaction. Plus précisément, pour tout comité possible, le représentant de chaque votant est le membre du comité qui est le mieux classé dans son ordre de préférences. La satisfaction de chaque agent est alors l'image par une fonction décroissante de la position de son représentant dans son ordre de préférences. Le comité optimal au sens de la règle de Chamberlin-Courant est le comité maximisant la somme des satisfactions des agents. Dans notre situation de vote, la règle de Chamberlin-Courant conduit donc forcément à l'élection du comité $\{1, 4, 5\}$*. Supposons maintenant que les préférences de nos agents sont représentables par les fonctions d'utilité suivantes :*

	1	2	3	4	5
u^1	0.35	0.30	0.25	0.10	0.00
u^2	0.00	0.30	0.25	0.35	0.10
u^3	0.10	0.30	0.25	0.00	0.35

Notons que ces valeurs numériques sont bien compatibles avec les relations $\succ_1$*,* $\succ_2$ *et* $\succ_3$*. Supposons aussi que chaque agent approuve un comité donné si et seulement si l'utilité de l'agent pour ce comité est plus grande que* 0.5*. Dans ce cas, aucun agent n'approuve le comité* $\{1, 4, 5\}$*, qui est élu par la règle de Chamberlin-Courant, alors que tous les comités comprenant* 2 *et* 3 *sont approuvés par tous.*

Cette sous-section s'organise comme suit : nous commençons par introduire le problème de sac à dos multi-agents avec vote par approbation avant d'étudier les enjeux computationnels liés à ce problème. Puis, nous proposons une procédure de recherche par séparation et évaluation permettant de déterminer

tous les gagnants possibles de ce problème de vote. Nous combinons ensuite cette procédure de recherche avec une stratégie d'élicitation incrémentale de manière à identifier un gagnant de l'élection. Enfin, nous terminons notre étude avec une analyse expérimentale de nos procédures.

4.2.1 Formalisation du problème et enjeux computationnels

Considérons un problème de décision collective où un ensemble d'agents $N = \{1, \ldots, n\}$ doivent choisir collectivement un sous-ensemble d'objets (e.g., candidats, projets) de l'ensemble $\mathcal{Q} = \{1, \ldots, q\}$. Les sous-ensembles $X \subseteq \mathcal{Q}$ réalisables sont ici caractérisés par la contrainte de sac à dos standard :

$$\sum_{j \in X} w_j \leq W$$

où $w_j > 0$ est le poids de l'objet j et $W > 0$ est le poids total maximum. Soulignons toutefois que les algorithmes proposés dans cette sous-section restent applicables après ajout de contraintes (linéaires) supplémentaires (e.g., budget, cardinalité, parité). Nous supposons ici que les préférences de l'agent $i, i \in N$, peuvent être représentées par une fonction $u^i : 2^{\mathcal{Q}} \to \mathbb{R}$ mesurant l'utilité globale de tout sous-ensemble $X \subseteq \mathcal{Q}$. Plus précisément, l'agent i préfère X à X' si et seulement si $u^i(X) \geq u^i(X')$. Dans le cadre du problème du sac à dos multi-agents, nous supposons que la fonction u^i est de la forme suivante :

$$u^i(X) = \sum_{j \in X} u^i_j$$

où $u^i_j \in \mathbb{R}$ représente l'utilité de l'objet j pour l'agent i. Les représentations numériques $u^i, i \in N$, des relations de préférence individuelle peuvent être utilisées dans le cadre du vote par approbation de la manière suivante : un sous-ensemble $X \subseteq \mathcal{Q}$ est approuvée par l'agent i si et seulement si $u^i(X) \geq \lambda^i$, où $\lambda^i \in \mathbb{R}$ est une valeur seuil, appelée *seuil d'approbation*, permettant de séparer les solutions approuvées de celles qui ne le sont pas. À partir des vecteurs $u = (u^1, \ldots, u^n)$ et $\lambda = (\lambda^1, \ldots, \lambda^n)$, nous pouvons définir le score d'approbation $\mathrm{A}(X, u, \lambda)$ de tout sous-ensemble réalisable $X \subseteq \mathcal{Q}$ comme suit :

$$\mathrm{A}(X, u, \lambda) = |\{i \in N : u^i(X) \geq \lambda^i\}|$$

Les sous-ensembles réalisables $X \subseteq \mathcal{Q}$ maximisant la valeur $\mathrm{A}(X, u, \lambda)$ sont les solutions optimales du problème. Dans un premier temps, nous nous intéressons à la complexité du problème de décision suivant :

Problème de Sac à dos Multi-agents avec vote par Approbation (PSMA) :

Entrée : un ensemble fini $\mathcal{Q}$ d'objets ; un poids total maximum $W > 0$; un poids $w_j > 0$ pour chaque objet $j \in \mathcal{Q}$; un ensemble fini N d'agents ; un seuil d'approbation $\lambda^i \in \mathbb{R}$ pour chaque agent $i \in N$; une utilité $u^i_j \in \mathbb{R}$ pour chaque agent $i \in N$ et chaque objet $j \in \mathcal{Q}$; un entier positif K.
Question : existe-t-il un ensemble $X \subseteq \mathcal{Q}$ tel que $\sum_{j \in X} w_j \leq W$ et $\mathrm{A}(X, u, \lambda) \geq K$?

Proposition 35. *PSMA est NP-complet.*

Démonstration. Cette proposition se montre très simplement à l'aide d'une réduction du problème de sac à dos standard (mono-agent). En effet, tester l'existence d'un sous-ensemble d'objets admissible avec une utilité plus grande K' pour un décideur donné est équivalent à résoudre une instance du problème PSMA avec $K = 1$ et un unique agent, celui-ci ayant les mêmes utilités sur les objets que le décideur et un seuil d'approbation égal à K'. □

Il existe toutefois des cas particuliers assez simples pour lesquels le problème devient plus facile, comme par exemple le suivant :

Proposition 36. *Si W est constant et si w_j est un entier pour tout $j \in \mathcal{Q}$, alors PSMA est dans $\mathcal{P}$.*

Démonstration. Puisque w_j est un entier strictement positif pour tout $j \in \mathcal{Q}$, alors nous savons que tous les sous-ensembles réalisables ont nécessairement au plus W objets. Le nombre de sous-ensembles de taille au plus W est :

$$\sum_{k=0}^{W} \binom{q}{k} = \sum_{k=0}^{W} \frac{q!}{k!(q-k)!}$$

Ce nombre est trivialement polynomial en q lorsque W est constant. Par conséquent, le résultat que nous souhaitons montrer peut être obtenu relativement facilement en considérant la procédure naïve suivante : pour tous les sous-ensembles $X \subseteq \mathcal{Q}$ de taille au plus W, tester si $\sum_{j \in X} w_j \leq W$ est vraie (condition de faisabilité) puis calculer le score $\mathrm{A}(X, u, \lambda) = |\{i \in N : u^i(X) \geq \lambda^i\}|$ dans le cas où le test réussit. Cette procédure, qui est polynomiale en q et n lorsque W est constant, permet en effet de déterminer s'il existe ou non une solution réalisable avec un score d'approbation supérieur ou égal à K. □

La proposition 35 montre que, dans le cas général, trouver le sous-ensemble d'objets maximisant le score d'approbation est NP-difficile. Par ailleurs, les méthodes de résolution pseudo-polynomiales fondées sur la programmation dynamique conçues pour le problème de sac à dos standard ne sont pas directement transposables au problème de sac à dos multi-agents avec vote par approbation :

Exemple 31. *Considérons un problème de décision collective où 2 agents doivent choisir ensemble 2 représentants parmi les 3 candidats qui se sont présentés (i.e. $N = \{1, 2\}$, $\mathcal{Q} = \{1, 2, 3\}$, $W = 2$ et $w_j = 1$ pour tout $j \in \mathcal{Q}$). Supposons que les utilités et les seuils d'approbation sont les suivants :*

u_1^1	u_2^1	u_3^1	λ^1	u_1^2	u_2^2	u_3^2	λ^2
0.5	0.3	0.2	0.5	0.2	0.5	0.3	0.7

Dans ce cas, nous avons $\mathrm{A}(\{1\}, u, \lambda) = 1 > 0 = \mathrm{A}(\{2\}, u, \lambda)$ *mais* $\mathrm{A}(\{1, 3\}, u, \lambda) = 1 < 2 = \mathrm{A}(\{2, 3\}, u, \lambda)$. *Nous observons donc une inversion du sens de préférence :* $\{1\}$ *est collectivement préféré à* $\{2\}$ *mais* $\{2, 3\}$ *est collectivement préféré à* $\{1, 3\}$. *Ainsi, les préférences collectives induites par les scores d'approbation ne sont pas additives par rapport à l'union de sous-ensembles disjoints. Ceci nous interdit de construire la solution optimale à partir de sous-ensembles optimaux pour le score d'approbation.*

Néanmoins, le sous-ensemble d'objets vainqueur de l'élection peut être déterminé en résolvant le programme linéaire en nombres entiers suivant (nommée LP_A) :

$$\max \quad \sum_{i \in N} a^i \tag{4.25}$$

$$\text{s.c.} \quad \sum_{j \in \mathcal{Q}} u_j^i x_j - \lambda^i \geq M(a^i - 1),\ \forall i \in N \tag{4.26}$$

$$\sum_{j \in \mathcal{Q}} w_j x_j \leq W \tag{4.27}$$

$$x_j \in \{0,1\}, a^i \in \{0,1\},\ \forall i \in N, \forall j \in \mathcal{Q}$$

Dans ce programme, le vecteur $x = (x_1, \ldots, x_q)$ représente le sous-ensemble $X \subseteq \mathcal{Q}$ contenant l'objet j si et seulement si $x_j = 1$. Par ailleurs, l'équation (4.27) correspond à la contrainte de sac à dos et M est une constante arbitrairement grande permettant d'introduire des variables booléennes $a^i, i \in N$, qui seront égales à 1 si et seulement si l'agent i approuve le sous-ensemble d'objets X associé au vecteur x. En effet, comme la fonction objectif est à maximiser, alors la variable a^i sera mise à 1 sauf si $u^i(X) - \lambda^i < 0$ (cf. équation 4.26), autrement dit sauf si l'agent i n'approuve pas le sous-ensemble X. Dans la suite, nous noterons $\mathcal{X}$ l'ensemble des vecteurs solutions $x = (x_1, \ldots, x_q) \in \{0,1\}^q$ représentant tous les sous-ensembles d'objets $X \subseteq \mathcal{Q}$ vérifiant la contrainte de sac-à-dos ; par définition, l'ensemble $\mathcal{X}$ est constitué de toutes les solutions réalisables de notre problème. Par abus de langage, nous utiliserons aussi les notations $u^i(x)$ et $\mathrm{A}(x, u, \lambda)$ pour faire référence à l'utilité $u^i(X)$ et au score $\mathrm{A}(X, u, \lambda)$ respectivement.

Déterminer un gagnant de l'élection en utilisant LP_A nécessite de connaître précisément les fonctions $u^i, i \in N$, ainsi que les seuils $\lambda^i, i \in N$. Cependant, l'élicitation totale des utilités individuelles et des seuils d'approbation est une tâche dont on doit faire l'économie lorsqu'il s'agit de prendre une décision en limitant les interventions des agents. C'est pourquoi nous nous intéressons dans cette sous-section à la résolution de ce problème par élicitation incrémentale de préférences. Comme étape préliminaire, nous étudions la problématique de recherche des gagnants possibles en présence de préférences incomplètes.

4.2.2 Détermination des gagnants possibles

Nous proposons ici un algorithme permettant de déterminer tous les gagnants possibles pour le vote par approbation en présence de préférences partiellement connues. Plus précisément, en entrée de notre algorithme, nous considérons trois ensembles de données préférentielles par agent $i \in N$:

- un ensemble A^i de solutions que l'agent i a approuvées,
- un ensemble $\bar{A}^i$ de solutions que l'agent i a désapprouvées,
- un ensemble $\mathcal{P}^i$ constitué de paires $(y, z) \in \mathcal{X}^2$ telles que nous savons que l'agent i préfère y à z.

Bien que les éléments de $\mathcal{P}^i$ ne soient pas des déclarations d'approbation, ces derniers peuvent être utilisés pour déduire de nouvelles informations en les combinant avec les données des ensembles A^i et $\bar{A}^i$. Par exemple, si $(y, z) \in \mathcal{P}^i$ et $z \in A^i$, alors nous savons que l'agent i approuve la solution y bien que cette information ne nous soit pas donnée de manière explicite.

Notons U^i (resp. Λ^i) l'ensemble des fonctions d'utilité u^i (resp. des seuils d'approbation λ^i) compatibles avec les informations disponibles sur les préférences de l'agent $i \in N$. Par définition, (U^i, Λ^i) est l'ensemble des paires (u^i, λ^i) vérifiant les inégalités suivantes :

1. $\forall y \in A^i$, $u^i(y) \geq \lambda^i$
2. $\forall y \in \bar{A}^i$, $u^i(y) < \lambda^i$
3. $\forall (y, z) \in \mathcal{P}^i$, $u^i(y) \geq u^i(z)$

où u^i est de la forme $u^i(a) = \sum_{j \in \mathcal{Q}} u^i_j a_j$ et $\lambda^i \in \mathbb{R}$. Étant donnés $U = U^1 \times \ldots \times U^n$ et $\Lambda = \Lambda^1 \times \ldots \times \Lambda^n$, il s'agit pour nous de déterminer l'ensemble $\mathrm{GP}(\mathcal{X}, U, \Lambda)$ des gagnants possibles pour le vote par approbation :

Définition 53. *L'ensemble $GP(\mathcal{X}, U, \Lambda)$ des gagnants possibles pour le vote par approbation est l'ensemble de toutes les solutions $x \in \mathcal{X}$ qui maximisent le score d'approbation pour au moins un profil d'utilités $u \in U$ et un vecteur de seuils d'approbation $\lambda \in \Lambda$. Plus formellement :*

$$GP(\mathcal{X}, U, \Lambda) = \bigcup_{(u,\lambda) \in U \times \Lambda} \arg\max_{x \in \mathcal{X}} \mathrm{A}(x, u, \lambda)$$

Rappelons que l'exemple 31 montre que les procédures de programmation dynamique standards ne peuvent pas être utilisées en l'état pour déterminer les gagnants de l'élection quand les utilités et les seuils d'approbation sont précisément connus. Cette difficulté demeure bien évidemment lorsque ces derniers sont imprécisément définis. De ce fait, nous nous intéressons plutôt à l'approche de résolution par séparation et évaluation (ou *branch and bound* en anglais), qui est l'outil générique le plus couramment utilisé pour résoudre des problèmes d'optimisation combinatoire NP-difficiles. Un algorithme par séparation et évaluation permet une énumération intelligemment guidée des solutions admissibles, exploitant des bornes sur la valeur optimale pour éviter d'explorer tout l'espace des solutions. Nous proposons maintenant un algorithme de recherche par séparation et évaluation permettant de déterminer l'ensemble $\mathrm{GP}(\mathcal{X}, U, \Lambda)$ des gagnants possibles pour le vote par approbation. Dans notre algorithme, les nœuds de l'arbre d'exploration représentent des instanciations partielles des variables de décision $x = (x_1, \ldots, x_q)$. Plus précisément, chaque nœud η de l'arbre d'exploration est caractérisé par une paire $(\mathcal{Q}^0_\eta, \mathcal{Q}^1_\eta)$, où $\mathcal{Q}^k_\eta = \{j \in \mathcal{Q} : x_j = k\}$ pour $k \in \{0, 1\}$. De manière moins formelle, l'ensemble $\mathcal{Q}^0_\eta$ (resp. $\mathcal{Q}^1_\eta$) représente l'ensemble des objets que nous avons rejetés (resp. sélectionnés) pour arriver au nœud η ; à la racine de l'arbre d'exploration, nous avons $\mathcal{Q}^0_\eta = \mathcal{Q}^1_\eta = \emptyset$. Chaque nœud η est associé à une région de l'espace des solutions. Plus précisément, la solution $x = (x_1, \ldots, x_q)$ est rattachée au nœud η si et seulement si :

1. $\sum_{j \in \mathcal{Q}} x_j w_j \leq W$ (contrainte de faisabilité)
2. $\forall k \in \{0, 1\}$, $\forall j \in \mathcal{Q}^k_\eta$, $x_j = k$

L'ensemble des solutions rattachées au nœud η sera noté S_η ci-après. Par ailleurs, nous noterons $\mathcal{Q}_\eta = \mathcal{Q} \setminus (\mathcal{Q}^0_\eta \cup \mathcal{Q}^1_\eta)$ l'ensemble des objets pour lesquels, au niveau du nœud η, il n'a pas encore été décidé s'ils devaient être sélectionnés ou non. Avec ces notations, nous pouvons à présent décrire les principales caractéristiques de notre procédure de recherche :

Initialisation. Une procédure de recherche par séparation et évaluation commence généralement par construire un ensemble S_0 de solutions réalisables à l'aide d'une méthode heuristique afin d'obtenir une borne initiale sur la valeur optimale du problème d'optimisation considéré. Pour le problème du sac à dos multi-agents avec vote par approbation, nous préconisons d'utiliser l'heuristique suivante : il s'agit tout d'abord de demander aux agents de trier les objets disponibles par ordre de préférences. Notons alors p_j^i la position de l'objet j dans l'ordre de préférences fournit par l'agent i. Définissons ensuite le score $\alpha_j^i = (q - p_j^i)/w_j$ pour chaque agent $i \in N$ et chaque objet $j \in \mathcal{Q}$ représentant le compromis réalisé par cet objet entre préférence et poids. Avec ces scores, nous construisons alors une solution s^i par agent $i \in N$ en utilisant un algorithme glouton consistant à sélectionner les objets $j \in \mathcal{Q}$ par ordre décroissant de valeur α_j^i, excluant ceux dont le poids excède la capacité résiduelle de la solution en construction. Pour chaque agent i, nous ajoutons à l'ensemble S_0 la solution s^i ainsi construite car celle-ci doit sûrement constituer une bonne solution du point de vue de l'agent i par définition. Par ailleurs, nous proposons d'appliquer cet algorithme glouton au score moyen défini par $\alpha_j = 1/n \sum_{i=1}^{n} \alpha_j^i$ pour tout objet $j \in \mathcal{Q}$. Cette dernière opération nous permet de compléter l'ensemble de solutions initial S_0 avec une solution un peu plus consensuelle que les solutions $s^i, i \in N$. Cette solution sera notée $\bar{s}$ dans la suite de cette sous-section.

Évaluation et élagage. Plaçons nous à une itération donnée de la procédure de recherche. Notons S l'ensemble des solutions réalisables détectées jusque-là (initialement, nous avons $S = S_0$). Notons par ailleurs O l'ensemble des nœuds de l'arbre qu'il reste à explorer. Notre règle d'élagage de nœuds est fondée sur la notion de *setwise max regret* que nous définissons comme suit :

Définition 54 (Setwise max Regret (SR))**.** *Le SR de $Y \subseteq \mathcal{X}$ par rapport à $Z \subseteq \mathcal{X}$ est défini par :*

$$SR(Y, Z, U, \Lambda) = \max_{(u,\lambda) \in U \times \Lambda} \left\{ \max_{z \in Z} \mathrm{A}(z, u, \lambda) - \max_{y \in Y} \mathrm{A}(y, u, \lambda) \right\}$$

Le *setwise max regret* $SR(Y, Z, U, \Lambda)$ de l'ensemble de solutions $Y \subseteq \mathcal{X}$ par rapport à l'ensemble de solutions $Z \subseteq \mathcal{X}$ est égal à la plus grande différence de score possible entre la meilleure solution de Z et la meilleure solution de Y. Par définition, si $SR(Y, Z, U, \Lambda) < 0$, alors Z ne contient aucun gagnant possible. En effet, de cette inégalité stricte, nous déduisons que, pour toute solution $z \in Z$, pour tout profil $u \in U$ et pour tout vecteur $\lambda \in \Lambda$, il existe une solution $y \in Y$ telle que $\mathrm{A}(z, u, \lambda) < \mathrm{A}(y, u, \lambda)$. Par conséquent, nous proposons d'élaguer tout nœud $\eta \in O$ tel que $SR(S, S_\eta, U, \Lambda) < 0$ car aucune solution rattachée au nœud η ne peut être un gagnant possible dans ce cas. Pour pouvoir calculer efficacement la valeur $SR(S, S_\eta, U, \Lambda)$, nous utilisons le résultat suivant :

$$\begin{aligned} SR(S, S_\eta, U, \Lambda) &= \max_{(u,\lambda) \in U \times \Lambda} \left\{ \max_{x \in S_\eta} \mathrm{A}(x, u, \lambda) - \max_{s \in S} \mathrm{A}(s, u, \lambda) \right\} \\ &= \max_{x \in S_\eta} \max_{(u,\lambda) \in U \times \Lambda} \min_{s \in S} \left\{ \mathrm{A}(x, u, \lambda) - \mathrm{A}(s, u, \lambda) \right\} \end{aligned}$$

Cette formulation alternative de la valeur $SR(S, S_\eta, U, \Lambda)$ nous permet de calculer $SR(S, S_\eta, U, \Lambda)$ en

résolvant le programme en nombres entiers suivant :

$$\max \quad t \tag{4.28}$$

$$\text{s.c.} \quad t \leq \sum_{i \in N} a^i - \sum_{i \in N} a^i_s, \ \forall s \in S \tag{4.29}$$

$$\sum_{j \in \mathcal{Q}^1_\eta} u^i_j + \sum_{j \in \mathcal{Q}_\eta} u^i_j x_j - \lambda^i \geq M(a^i - 1), \forall i \in N \tag{4.30}$$

$$\sum_{j \in \mathcal{Q}} u^i_j s_j - \lambda^i + \xi \leq M a^i_s, \ \forall s \in S, \ \forall i \in N \tag{4.31}$$

$$\sum_{j \in \mathcal{Q}_\eta} w_j x_j + \sum_{j \in \mathcal{Q}^1_\eta} w_j \leq W \tag{4.32}$$

$$\sum_{j \in \mathcal{Q}} u^i_j = M, \ \forall i \in N \tag{4.33}$$

$$\sum_{j \in \mathcal{Q}} u^i_j y_j \geq \lambda^i, \ \forall i \in N, \forall y \in A^i \tag{4.34}$$

$$\sum_{j \in \mathcal{Q}} u^i_j y_j \leq \lambda^i - \xi, \ \forall i \in N, \forall y \in \bar{A}^i \tag{4.35}$$

$$\sum_{j \in \mathcal{Q}} u^i_j y_j \geq \sum_{j \in \mathcal{Q}} u^i_j z_j, \ \forall i \in N, \forall (y, z) \in \mathcal{P}^i \tag{4.36}$$

$$u^i_j \geq 0, \lambda^i \geq 0, \ \forall i \in N, \forall j \in \mathcal{Q} \tag{4.37}$$

$$t \in \mathbb{R}; \ x_j \in \{0, 1\}, \ \forall j \in \mathcal{Q}_\eta$$

$$a^i \in \{0, 1\}, \ \forall i \in N; \ a^i_s \in \{0, 1\}, \ \forall i \in N, \forall s \in S$$

Dans ce programme, l'équation (4.32) correspond à la contrainte de sac à dos et $\xi > 0$ est une constante arbitrairement petite permettant de modéliser les inégalités strictes. Les équations (4.34-4.36) contraignent les utilités et seuils d'approbation à être compatibles avec les préférences collectées. Les équations (4.33) et (4.37) sont des contraintes de normalisation. En effet, sans perte de généralité, nous pouvons supposer que les utilités u^i et les seuils λ^i sont positifs et bornées supérieurement par une constante M puisque les préférences ne sont pas modifiées après application d'une fonction affine strictement croissante. En ce qui concerne les variables booléennes $a^i, i \in N$, celles-ci seront mises à 1 si et seulement si l'agent i approuve la solution x. En effet, comme la variable t est à maximiser et que t doit être inférieure à $\sum_{i \in N} a^i - \sum_{i \in N} a^i_s$ pour tout $s \in S$ (cf. équation (4.29)), alors chaque variable a^i sera mise à 1 sauf si celle-ci est contrainte de valoir 0 par l'équation (4.30). Cette dernière équation impose que la variable a^i soit mise à 0 dans le cas où $\sum_{j \in \mathcal{Q}^1_\eta} u^i_j + \sum_{j \in \mathcal{Q}_\eta} u^i_j x_j < 0$, i.e. lorsque que la solution x n'est pas approuvée par l'agent i. Ainsi, la variable a^i est égale à 1 si et seulement si la solution x est approuvée par l'agent i. De manière analogue, pour tout $i \in N$ et tout $s \in S$, la variable booléenne a^i_s sera mise à 0 pour maximiser la fonction objectif sauf si celle-ci est contrainte à valoir 1 par l'équation (4.31). Cette équation oblige la variable a^i_s à être égale à 1 si et seulement si $\sum_{j \in \mathcal{Q}} u^i_j s_j - \lambda^i \geq 0$, i.e. si et seulement si la solution s est approuvée par l'agent i. Ceci nous permet de conclure que l'équation (4.29)

introduit une variable $t \in \mathbb{R}$ représentant, pour tout $(u, \lambda) \in U \times \Lambda$ et pour tout $x \in S_\eta$, la différence de score d'approbation entre la solution x et la meilleure solution de l'ensemble S. Ainsi, ce programme en nombres entiers nous permet bien de calculer la valeur $SR(S, S_\eta, U, \Lambda)$ pour un nœud η. Ce programme contient une variable u_j^i par couple $(i, j) \in N \times \mathcal{Q}$, une variable a_s^i par couple $(i, s) \in N \times S$, une variable λ^i et une variable a^i par agent $i \in N$, une variable de décision x_j par objet $j \in \mathcal{Q}_\eta$ et enfin une variable réelle t ; nous avons donc en tout $n(q + |S| + 2) + |\mathcal{Q}_\eta| + 1$ variables. Notons toutefois que ce programme n'est pas linéaire car l'équation (4.30) inclut des termes quadratiques de type $u_j^i x_j$ avec $i \in N$ et $j \in \mathcal{Q}_\eta$. Afin de linéariser les contraintes associées à cette équation, nous introduisons des variables positives v_j^i permettant de représenter le produit $u_j^i x_j$ pour tout $i \in N$ et tout $j \in \mathcal{Q}_\eta$. Plus précisément, il suffit de remplacer l'équation (4.30) par les contraintes suivantes :

$$\sum_{j \in \mathcal{Q}_\eta^1} u_j^i + \sum_{j \in \mathcal{Q}_\eta} v_j^i - \lambda^i \geq M(a^i - 1),\ \forall i \in N$$

$$v_j^i \leq u_j^i,\ \forall i \in N, \forall j \in \mathcal{Q}_\eta \tag{4.38}$$

$$v_j^i \leq M x_j,\ \forall i \in N, \forall j \in \mathcal{Q}_\eta \tag{4.39}$$

$$v_j^i \geq u_j^i + M(x_j - 1),\ \forall i \in N, \forall j \in \mathcal{Q}_\eta \tag{4.40}$$

$$v_j^i \geq 0,\ \forall i \in N, \forall j \in \mathcal{Q}_\eta$$

Les quatre dernières équations permettent de garantir que chaque variable v_j^i est bien égale au produit $u_j^i x_j$. En effet, dans le cas où $x_j = 1$, nous avons d'une part $v_j^i \leq \min\{u_j^i, M\} = u_j^i$ par les équations (4.38-4.39) et d'autre part $v_j^i \geq u_j^i$ par l'équation (4.40). Par conséquent, si $x_j = 1$, alors nous avons forcément $v_j^i = u_j^i = u_j^i x_j$. Dans le cas où $x_j = 0$, nous avons obligatoirement $v_j^i = 0$ puisque v_j^i est une variable positive et que l'équation (4.39) impose la contrainte $v_j^i \leq M \times 0 = 0$. Signalons que cette instanciation de la variable v_j^i n'entre en contradiction ni avec l'équation (4.38) (car u_j^i est une variable positive) ni avec l'équation (4.39) (car $u_j^i - M \leq 0$ par définition de la constante M). De ce fait, nous avons bien $v_j^i = u_j^i x_j$ dans tous les cas. Le programme linéaire en nombres entiers résultant de cette modification sera appelé PLNE_η ci-après.

Séparation. Les valeurs $SR(S, S_\eta, U, \Lambda)$ disponibles aux différents nœuds $\eta \in O$ sont utilisées non seulement pour élaguer des nœuds durant l'exploration, mais aussi pour sélectionner le prochain nœud à explorer. Plus précisément, nous choisissons d'explorer en priorité un nœud $\eta^* \in O$ maximisant la valeur $SR(S, S_\eta, U, \Lambda)$. Cette stratégie a pour but d'améliorer le plus possible nos capacités d'élagage, en insérant le plus rapidement possible de meilleures solutions dans l'ensemble de solutions S. En effet, par définition, la solution optimale x^* de PLNE_{η^*} vérifie :

$$\max_{(u,\lambda) \in U \times \Lambda} \Big\{ \mathrm{A}(x^*, u, \lambda) - \max_{s \in S} \mathrm{A}(s, u, \lambda) \Big\} \geq \max_{(u,\lambda) \in U \times \Lambda} \Big\{ \mathrm{A}(x, u, \lambda) - \max_{s \in S} \mathrm{A}(s, u, \lambda) \Big\}$$

pour tout $x \in \bigcup_{\eta \in O} S_\eta$. L'exploration du nœud $\eta^* \in O$ consiste à diviser/séparer l'ensemble de solutions

S_{η^*} en deux sous-ensembles disjoints, en suivant les instanciations possibles d'une variable x_j de PLNE$_{\eta^*}$. Nous proposons de choisir en priorité un objet $j \in \mathcal{Q}_\eta$ tel que $x_j^* = 1$ dans l'optique d'insérer le plus rapidement possible la solution x^* dans S.

Filtrage. Dans la partie "évaluation et élagage", nous avons vu que notre règle d'élagage supprime uniquement des nœuds η vérifiant $S_\eta \cap \mathrm{GP}(\mathcal{X}, U, \Lambda) = \emptyset$. Par conséquent, notre algorithme par séparation et évaluation retourne, en général, un sur-ensemble des gagnants possibles pour le vote par approbation. Afin de retirer de l'ensemble S les éléments indésirables, nous appliquons à la fin de son exécution une procédure de filtrage sur l'ensemble S. Cette procédure, nommée (U, Λ)-Filter, consiste à supprimer itérativement toutes les solutions $s' \in S$ telles que $SR(S \backslash \{s'\}, \{s'\}, U, \Lambda) < 0$ (de manière analogue à notre procédure Ω-Filter en décision multicritère). Chaque test est réalisé à l'aide du programme suivant :

$$
\begin{aligned}
\max\ & t \\
\text{s.c.}\ & \text{Équation (4.29)} \\
& \sum_{j \in P} u_j^i s'_j - \lambda^i \geq M(a^i - 1),\ \forall i \in N \\
& \text{Équations (4.31-4.37)} \\
& a^i \in \{0,1\},\ \forall i \in N;\ a_s^i \in \{0,1\},\ \forall i \in N, \forall s \in S
\end{aligned}
$$

Pour résumer, ces différentes opérations sont mises en œuvre dans l'algorithme 8.

Algorithm 8: Recherche fondée sur les scores d'approbation

Input: S_0 : solutions initiales ; $A^i, \bar{A}^i, \mathcal{P}^i$: préférences collectées auprès de l'agent i

$S \leftarrow S_0$; $\eta \leftarrow [\emptyset, \emptyset]$; $O \leftarrow \{\eta\}$
while $O \neq \emptyset$ **do**
  Choisir un nœud η^* dans l'ensemble $\arg\max_{\eta \in O} SR(S, S_\eta, U, \Lambda)$
  if $\mathcal{Q}_{\eta^*} = \emptyset$ **then**
    $S \leftarrow S \cup S_{\eta^*}$
  else
    Choisir un objet $j \in \mathcal{Q}_\eta$ si possible tel que x_j vaut 1 dans la solution optimale de PLNE$_\eta$
    Générer les nœuds $\eta_0^* = [\mathcal{Q}_{\eta*}^0 \cup \{j\}, \mathcal{Q}_{\eta^*}^1]$ et $\eta_1^* = [\mathcal{Q}_{\eta^*}^0, \mathcal{Q}_{\eta^*}^1 \cup \{j\}]$
    forall $\eta \in \{\eta_0^*, \eta_1^*\}$ **do**
      if $S_\eta \neq \emptyset$ *et* $SR(S, S_\eta, U, \Lambda) \geq 0$ **then**
        $O \leftarrow O \cup \{\eta\}$
      end
    end
  end
  $O \leftarrow O \setminus \{\eta^*\}$
end
return (U, Λ)-Filter(S)

Cet algorithme est justifié par la proposition suivante :

Proposition 37. *L'algorithme 8 retourne l'ensemble GP*$(\mathcal{X}, U, \Lambda)$.

Démonstration. Puisque notre règle d'élagage ne supprime que des nœuds ne contenant aucun gagnant possible, alors S est un sur-ensemble de $\mathrm{GP}(\mathcal{X}, U, \Lambda)$ à la fin de la boucle "while". Supposons qu'il existe $s' \in S$ tel que $s' \notin \mathrm{GP}(\mathcal{X}, U, \Lambda)$ à la fin de la boucle. Puisque s' n'est pas un gagnant possible, alors pour tout $(u, \lambda) \in U \times \Lambda$, il existe $s \in \mathrm{GP}(\mathcal{X}, U, \Lambda) \subseteq S$ tel que $\mathrm{A}(s, u, \lambda) < \mathrm{A}(s', u, \lambda)$. Par conséquent, la solution s' sera forcément supprimée durant le filtrage par définition de ce dernier. Ainsi, seules les solutions de $\mathrm{GP}(\mathcal{X}, U, \Lambda)$ sont retournées à la fin. □

En fonction des ensembles U^i et Λ^i définis implicitement par les données de préférences collectées auprès de l'agent i, il est possible que l'ensemble $\mathrm{GP}(\mathcal{X}, U, \Lambda)$ des gagnants possibles soit trop grand pour être énumérer de manière efficace. Afin de réduire les temps de résolution, nous pouvons envisager un instant de déterminer une approximation de cet ensemble avec garantie de performance. Rappelons qu'un FPTAS est un algorithme d'approximation ayant une complexité en temps et en espace qui est polynomiale à la fois en la taille de l'instance et en $1/\varepsilon$, où ε est le paramètre représentant l'erreur d'approximation. Dans la littérature, plusieurs FPTAS ont été proposées pour le calcul d'une approximation des solutions dans le cadre du problème de sac à dos standard (e.g., [Ibarra and Kim, 1975, Lawler, 1979]). Malheureusement, ces algorithmes ne s'étendent pas directement au problème de sac à dos multi-agents avec vote par approbation car ces derniers exploitent des algorithmes de programmation dynamique standards, conçus pour le problème du sac à dos mono-agent. À la place, nous proposons maintenant un algorithme de résolution interactif permettant de déterminer une solution nécessairement optimale par élicitation incrémentale des préférences.

4.2.3 Un algorithme interactif par séparation et évaluation pour la détermination d'un gagnant de l'élection

Dans la sous-section précédente, nous avons proposé un algorithme pour la détermination des gagnants possibles pour le vote par approbation (cf. algorithme 8). Cependant, en présence de préférences incomplètes, le nombre de gagnants possibles peut être très grand du fait de la nature combinatoire de l'ensemble des solutions admissibles. Cet algorithme peut donc être utilisé uniquement dans des situations de vote où une portion relativement importante des jugements d'approbation est disponible. Dans les autres situations, il est plus raisonnable d'interroger les agents pour diminuer l'imprécision qui entoure leurs préférences de manière à pouvoir prendre une décision.

Dans l'idéal, les ensembles U et Λ devraient réduire suffisamment pour permettre l'identification d'un *gagnant nécessaire* : une solution $x \in \mathcal{X}$ est un gagnant nécessaire si et seulement si $\mathrm{A}(x, u, \lambda) \geq \mathrm{A}(y, u, \lambda)$ pour toute solution $y \in \mathcal{X}$ et toute paire $(u, \lambda) \in U \times \Lambda$. Notons que si, à la fin de la recherche réalisée par l'algorithme 8, il existe une solution $x^* \in S$ telle que $\mathrm{A}(x^*, u, \lambda) \geq \mathrm{A}(y, u, \lambda)$ pour toute solution $y \in S$ et toute paire $(u, \lambda) \in U \times \Lambda$, alors la solution x^* est un gagnant nécessaire. En effet, rappelons que l'ensemble S contient tous les gagnants possibles à la fin de la recherche. Par conséquent, pour tout

$y \in \mathcal{X}$ et pour tout $(u, \lambda) \in U \times \Lambda$, il existe $x \in S$ tel que $\mathrm{A}(x, u, \lambda) \geq \mathrm{A}(y, u, \lambda)$ ce qui implique $\mathrm{A}(x^*, u, \lambda) \geq \mathrm{A}(y, u, \lambda)$ par transitivité. Ce résultat pourrait nous insuffler l'envie de mettre en œuvre une méthode en deux phases consistant tout d'abord à appliquer l'algorithme 8 puis à poser des questions aux agents dans le but de révéler une alternative $x^* \in S$ telle que $\mathrm{A}(x^*, u, \lambda) \geq \mathrm{A}(y, u, \lambda)$ pour toute solution $y \in S$ et toute paire $(u, \lambda) \in U \times \Lambda$. Cependant, une telle procédure ne serait pas très efficace en pratique car l'ensemble des gagnants possibles peut être très grand quand les ensembles U et Λ initiaux ne sont pas suffisamment discriminants. À la place, nous proposons une nouvelle fois de combiner la recherche et l'élicitation incrémentale des préférences de sorte à concentrer plus rapidement nos efforts sur des régions pertinentes de l'espace de solutions et à réduire le nombre d'interactions avec les agents en posant uniquement des questions à des moments clés de la recherche. Intégrer l'élicitation à la recherche effectuée par l'algorithme 8 est rendue possible par le résultat suivant :

$$\forall U' \subseteq U, \forall \Lambda' \subseteq \Lambda, \forall Y, Z \subset \mathcal{X},\ SR(Y, Z, U', \Lambda') \leq SR(Y, Z, U', \Lambda')$$

Ainsi, lorsque U et Λ réduisent en cours de résolution, nous sommes sûrs que tout élagage ne sera jamais remis en cause plus tard durant la recherche. Plus précisément, l'algorithme retournera à la fin l'ensemble $\mathrm{GP}(\mathcal{X}, U_f, \Lambda_f)$, où U_f et Λ_f représentent les ensembles U et Λ à la fin de la résolution. À présent, il reste à déterminer quand poser des questions durant la recherche pour garantir l'existence d'une alternative $x^* \in S$ telle que l'inégalité $\mathrm{A}(x^*, u, \lambda) \geq \mathrm{A}(y, u, \lambda)$ soit vraie pour toute solution $y \in S$ et toute paire $(u, \lambda) \in U \times \Lambda$ à la fin de l'exécution.

Notre procédure consiste à poser des questions durant l'exécution de l'algorithme 8 de manière à discriminer entre les solutions trouvées jusque là (celles qui sont stockées dans S). En pratique, l'ensemble S est maintenant restreint aux solutions qui ont reçu le plus d'approbations. Parmi ces solutions, nous choisissons arbitrairement un représentant, que nous appelons le *titulaire*. Initialement, le titulaire peut être n'importe quelle solution du problème de sac à dos, comme par exemple la solution de consensus $\bar{s}$ qui a été introduite dans la partie initialisation de la sous-section précédente. Durant l'exploration, à chaque fois qu'une nouvelle solution est trouvée (i.e. lorsque le test en ligne 6 de l'algorithme 8 réussit), celle-ci est comparée au titulaire en terme de scores d'approbation. Nous expliquons ci-après comment cette comparaison est mise en œuvre.

Étant donnés des ensembles U et Λ, une solution $x \in \mathcal{X}$ est nécessairement approuvée par un agent $i, i \in N$, si l'inégalité $u^i(x) \geq \lambda^i$ est vraie pour tout $(u, \lambda) \in U \times \Lambda$. Cette dernière propriété est en fait vérifiée si et seulement si la valeur optimale du programme linéaire suivant est positive :

$$\begin{aligned} \min \quad & \sum_{j \in \mathcal{Q}} u^i_j x_j - \lambda^i \\ \text{s.c.} \quad & \text{Équations (4.33-4.37)} \end{aligned}$$

Par conséquent, tester si une solution $x \in \mathcal{X}$ est nécessairement approuvée par un agent $i, i \in N$, peut être réalisé en temps polynomial par programmation linéaire. De même, tester si une solution $x \in \mathcal{X}$

est nécessairement désapprouvée par un agent $i, i \in N$, peut être effectué en utilisant la programmation linéaire. Ainsi, à chaque fois qu'une nouvelle solution est trouvée (appelée *challenger*), il est possible de déterminer de manière efficace si des agents approuvent/désapprouvent nécessairement cette solution. Par conséquent, une stratégie d'élicitation qui semble naturelle consiste à demander à tous les autres agents s'ils approuvent ou non le challenger. De cette façon, nous pouvons dériver le score d'approbation du challenger et le comparer avec celui du titulaire. Le challenger est ajoutée à l'ensemble S si celui-ci a le même score d'approbation que le titulaire. Si en revanche le challenger a un score d'approbation strictement plus grand que celui du titulaire, alors l'ensemble S est réinitialisé de sorte à contenir uniquement le challenger, ce dernier devenant le nouveau titulaire. Rien ne se passe dans le cas où le challenger reçoit strictement moins de votes que le titulaire. Cette stratégie d'élicitation intégrée à la résolution nous donne un algorithme de recherche interactif que nous nommerons RIA (pour Recherche Interactive fondée sur les scores d'Approbation) ci-après.

L'algorithme RIA permet de déterminer un gagnant de l'élection par élicitation incrémentale de préférences. En réalité, l'algorithme RIA peut être interrompu à tout moment de la résolution et retourner la meilleure solution trouvée jusque là (c'est-à-dire le titulaire). Supposons que l'algorithme a été arrêté à une certaine itération k. Notons s_k le titulaire à la fin de cette itération. Il est intéressant de remarquer que la qualité de la solution s_k peut être mesurée à l'aide de la notion de *Max Regret* suivante :

$$MR(s_k, \mathcal{X}, U_k, \Lambda_k) = \max_{x \in \mathcal{X}} \max_{(u,\lambda) \in U_k \times \Lambda_k} \Big\{ \mathrm{A}(x, u, \lambda) - \mathrm{A}(s_k, u, \lambda) \Big\}$$

où U_k et Λ_k représentent respectivement l'état des ensembles U et Λ à la fin de l'itération k. Ce regret est en effet une borne supérieure de l'écart à l'optimalité. Plus précisément, cette quantité est égale à la plus grande perte possible en terme de score d'approbation lorsque la solution s_k est recommandée aux agents. Cette valeur peut être calculé relativement facilement en utilisant les valeurs $SR(\{s_k\}, S_\eta, U_k, \Lambda_k)$ qui sont disponibles aux différents nœuds $\eta \in O_k$, où O_k représente l'état de l'ensemble O à la fin de l'itération k. Plus précisément, nous avons le résultat suivant :

Proposition 38. $MR(s_k, \mathcal{X}, U_k, \Lambda_k) = \max_{\eta \in O_k} SR(\{s_k\}, S_\eta, U_k, \Lambda_k)$.

Démonstration. Montrons que nous avons $\max_{\eta \in O_k} SR(\{s_k\}, S_\eta, U_k, \Lambda_k) = \max_{x \in \mathcal{X}} R(s_k, x, U_k, \Lambda_k)$, où $R(s_k, x, U_k, \Lambda_k) = \max_{(u,\lambda) \in U_k \times \Lambda_k} \{\mathrm{A}(x, u, \lambda) - \mathrm{A}(s_k, u, \lambda)\}$. Soit Y l'ensemble de solutions défini par $Y = \{x \in S_\eta : \eta \in O_k\}$. Soulignons le fait que l'ensemble Y n'est pas vide puisque l'algorithme RIA a été interrompu. Par définition des regrets SR, nous avons pour tout noeud $\eta \in O_k$:

$$\begin{aligned}
SR(\{s_k\}, S_\eta, U_k, \Lambda_k) &= \max_{(u,\lambda) \in U_k \times \Lambda_k} \Big\{ \max_{x \in S_\eta} \{\mathrm{A}(x, u, \lambda)\} - \mathrm{A}(s_k, u, \lambda) \Big\} \\
&= \max_{(u,\lambda) \in U_k \times \Lambda_k} \max_{x \in S_\eta} \{\mathrm{A}(x, u, \lambda) - \mathrm{A}(s_k, u, \lambda)\} \\
&= \max_{x \in S_\eta} \max_{(u,\lambda) \in U_k \times \Lambda_k} \{\mathrm{A}(x, u, \lambda) - \mathrm{A}(s_k, u, \lambda)\} \\
&= \max_{x \in S_\eta} R(s_k, x, U_k, \Lambda_k)
\end{aligned}$$

Par conséquent, nous avons $\max_{\eta \in O_k} SR(\{s_k\}, S_\eta, U_k, \Lambda_k) = \max_{x \in Y} R(s_k, x, U_k, \Lambda_k)$. Par ailleurs, par définition de notre règle d'élagage, nous avons $SR(\{s_k\}, S_\eta, U_k, \Lambda_k) \geq 0$ pour tout $\eta \in O_k$, ce qui implique $\max_{x \in Y} R(s_k, x, U_k, \Lambda_k) \geq 0$. En conséquence, afin d'établir le résultat souhaité, il suffit de montrer que nous avons $R(s_k, x, U_k, \Delta_k) \leq 0$ pour toute solution $x \in \mathcal{X} \backslash Y$. Trois cas sont alors possibles :

- *Cas où $x = s_k$:* nous avons trivialement $R(s_k, x, U_k, \Lambda_k) = 0$ dans ce cas.
- *Cas où $x \in S$:* dans ce cas, puisque s_k est le *titulaire* à la fin de l'itération k, alors $\mathrm{A}(x, u, \lambda) = \mathrm{A}(s_k, u, \lambda)$ pour tout $(u, \lambda) \in U_k \times \Lambda_k$ (par définition de notre stratégie d'élicitation). Par conséquent, nous avons forcément $R(s_k, x, U_k, \Lambda_k) = 0$.
- *Cas où $x \in S_\eta$, où η est un nœud qui a été élagué durant la recherche :* soit s' le *titulaire* qui a permis d'élaguer le nœud η. Par définition de la règle d'élagage, nous avons $SR(\{s'\}, S_\eta, U', \Lambda') < 0$, où U' et Λ' représentent respectivement l'état des ensembles U et Λ au moment de l'élagage. Comme $U_k \subset U'$ et $\Lambda_k \subseteq \Lambda'$, alors nous en déduisons $SR(\{s'\}, S_\eta, U_k, \Lambda_k) < 0$. Par ailleurs, par définition de notre stratégie d'élicitation, nous savons que $\mathrm{A}(s_k, u, \lambda) \geq \mathrm{A}(s', u, \lambda)$ pour tout $u \in U_k$ et tout $\lambda \in \Lambda_k$. Par conséquent, nous avons $SR(\{s_k\}, S_\eta, U_k, \Lambda_k) \leq SR(\{s'\}, S_\eta, U_k, \Lambda_k) < 0$. Enfin, comme nous venons de montrer que $SR(\{s_k\}, S_\eta, U_k, \Lambda_k) = \max_{x' \in S_\eta} R(s_k, x', U_k, \Lambda_k)$, alors nous en concluons $R(s_k, x, U_k, \Lambda_k) < 0$.

Ainsi, nous avons montré $MR(s_k, \mathcal{X}, U_k, \Lambda_k) = \max_{x \in \mathcal{X}} R(s_k, x, U_k, \Lambda_k) = \max_{x \in Y} R(s_k, x, U_k, \Lambda_k) = \max_{\eta \in O_k} SR(\{s_k\}, S_\eta, U_k, \Lambda_k)$. □

4.2.4 Résultats expérimentaux

Comme déterminer le sous-ensemble d'objets maximisant le score d'approbation est un problème NP-difficile, et ce même lorsque les utilités et les seuils d'approbation sont connus (cf. proposition 35), alors toute procédure par séparation et évaluation pour résoudre ce problème peut bien évidemment avoir de très mauvais temps de résolution (dans le pire cas, toutes les solutions peuvent être énumérées). Communément, l'efficacité d'une procédure par séparation et évaluation est estimée de manière empirique, en analysant des instances avec un grand nombre de solutions possibles. Dans cette sous-section, nous reportons des résultats d'expériences permettant de démontrer l'efficacité pratique de nos algorithmes. Les temps de calcul sont ici donnés en secondes, pour des expériences réalisées sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire. Dans ces expériences, nous utilisons le solveur Gurobi depuis un programme écrit en Java pour résoudre tous les programmes linéaires intervenant dans les calculs.

Tout d'abord, nous nous intéressons aux performances de l'algorithme 8 conçu pour la problématique de détermination des gagnants possibles ; cet algorithme sera appelé RA (pour Recherche fondée sur les scores d'Approbation) ci-après. Afin d'évaluer les performances de RA, nous considérons des instances du problème de sac à dos multi-agents avec $q = 12$ et dont les poids $w_j, j \in \mathcal{Q}$, ont été tirés uniformément dans l'ensemble $\{1, \ldots, 100\}$. Dans ces expériences, la capacité du sac à dos W est quant à elle définie par $W = d \times \sum_{j \in \mathcal{Q}} w_j$, où $d = 0.3, 0.4$ et 0.5 de manière à faire varier le nombre de solutions admissibles. De cette façon, en moyenne, nous obtenons des instances avec respectivement $500, 1000$ et 2000 sous-ensembles d'objets qui sont à la fois admissibles et maximaux par rapport à l'inclusion. Par ailleurs, pour

évaluer l'impact des ensembles U et Λ sur les temps de résolution, nous posons $p = 10, 20$ questions par agent (engendrées aléatoirement) avant de lancer la recherche. Les moyennes des temps obtenus sur 30 tests, pour des problèmes impliquant 20 agents, sont données dans la table 4.2. Dans cette table, nous reportons aussi la taille de l'ensemble retourné par l'algorithme RA.

	$d = 0.3$		$d = 0.4$		$d = 0.5$	
p	temps	$\#S$	temps	$\#S$	temps	$\#S$
10	26.6	19.8	71.8	29.8	614.4	111.4
20	25.2	17.4	55.7	23.9	442.3	90.3

TABLE 4.2 – Performances de l'algorithme RA (20 agents, 30 tests).

Dans la table 4.2, nous voyons tout d'abord que les temps de calcul augmentent rapidement avec la taille du problème. Par ailleurs, nous observons que les temps de résolution diminuent lentement avec le nombre de données collectées avant de lancer la recherche (lorsque celles-ci ont été engendrées aléatoirement). Toutefois, notre algorithme est capable de déterminer tous les gagnants possibles en moins de 10 minutes sur toutes les instances considérées.

À présent, nous souhaitons évaluer les performances de RIA notre algorithme de recherche interactive. Dans les expériences qui vont suivre, nous supposons connaître initialement les préférences des agents sur les objets pris séparément, pour pouvoir construire l'ensemble de solutions initiales S_0 (comme décrit en Section 4.2.2). Par ailleurs, les réponses aux questions vont être simulées à l'aide de fonctions d'utilité et de seuils d'approbation engendrés aléatoirement avec des corrélations négatives entre les agents, de sorte à obtenir des instances compliquées. Les premiers tests ont pour objectif d'évaluer les performances de la règle d'élagage de RIA en terme de nœuds explorés en moyenne (ce qui correspond à la taille moyenne de l'arbre d'exploration). Comme point de comparaison, nous reportons aussi le nombre de nœuds explorés par une énumération exhaustive des solutions réalisables (cette procédure naïve sera nommée S0 ci-après). Du fait que RIA puisse collecter des données de préférence, cet algorithme est capable de résoudre des instances bien plus grandes que les précédentes. Nous considérons ici des instances avec $q = 12, 15, 18, 20$ (et $d = 0.4$), conduisant à des problèmes de décision collective avec respectivement 1000, 8500, 60000 et 250000 sous-ensembles d'objets admissibles et maximaux en moyenne (cf. table 4.3).

q	S0	RIA
12	3024	109
15	21468	238
18	152415	289
20	647834	440

TABLE 4.3 – Nombre de nœuds explorés en moyenne (30 agents, 30 tests).

Dans la table 4.3, nous observons que le nombre de nœuds explorés par RIA reste relativement faible lorsque le nombre de solutions admissibles croît, ce qui est loin d'être le cas pour l'énumération exhaustive. Par exemple, notre règle d'élagage permet de réduire la taille moyenne de l'arbre d'exploration par un facteur 30 pour $q = 12$ et par un facteur 1500 pour $q = 20$.

Nous souhaitons à présent évaluer la pertinence de la stratégie de séparation de RIA, car celle-ci impacte directement l'efficacité de sa stratégie d'élicitation. En effet, rappelons que cette stratégie d'élicitation consiste à demander aux agents s'ils approuvent ou non des solutions qui ont été découvertes durant la recherche ; ces solutions dépendent de la stratégie de séparation choisie. À titre de comparaison, nous considérons aussi la procédure par séparation et évaluation (nommée Random ci-après) qui diffère de RIA uniquement sur la stratégie de séparation : nous choisissons aléatoirement le prochain nœud à explorer ainsi que la prochaine variable à instancier. Pour les deux procédures considérées, nous calculons le regret MR du titulaire courant (en utilisant la proposition 38) à chaque fois qu'un agent répond à une question durant la recherche. Rappelons que ce regret est une borne supérieure de l'erreur commise en recommandant le titulaire (en terme de score d'approbation). En particulier, lorsque ce regret est égal à 0, alors le titulaire courant est un gagnant nécessaire. Les regrets sont ici exprimés sur une échelle normalisée attribuant la valeur 1 au regret initial, c'est-à-dire avant d'avoir posé des questions. La figure 4.6 présente les moyennes des résultats obtenus sur 30 tests.

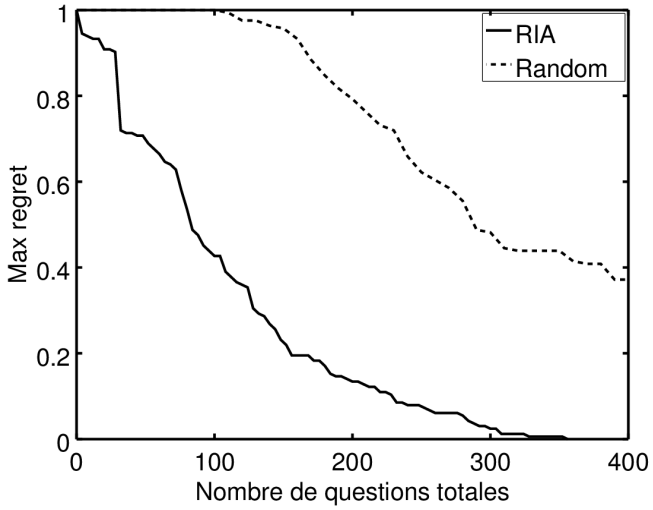


FIGURE 4.6 – MR du titulaire par rapport au nombre de questions posées ($q = 12$, $d = 0.4$, $n = 30$).

Dans la figure 4.6, nous voyons que la valeur MR du titulaire réduit beaucoup plus rapidement avec l'algorithme RIA qu'avec l'algorithme Random. Par exemple, après 240 questions (c'est-à-dire 8 questions par agent en moyenne), le regret associé au choix du titulaire est en dessous de 10% du regret initial avec RIA, tandis que ce regret est toujours au dessus de 65% avec la stratégie de séparation aléatoire. Ainsi, notre stratégie d'élicitation semble être bien plus efficace avec notre stratégie de séparation fondée sur les regrets que lorsque celle-ci est combinée avec la stratégie de séparation aléatoire.

Les tests suivants ont pour but d'évaluer la procédure RIA en tant qu'algorithme interruptible à tout moment (ou *anytime algorithm* en anglais). À titre de comparaison, nous considérons une nouvelle fois la procédure Random. Pour ce faire, nous calculons la valeur MR du titulaire à chaque itération des procédures de résolution, ce qui nous donne une garantie sur l'erreur associée à la recommandation du titulaire à chaque itération de celles-ci. Les moyennes des valeurs MR obtenues sur 30 tests sont données dans la figure 4.7.

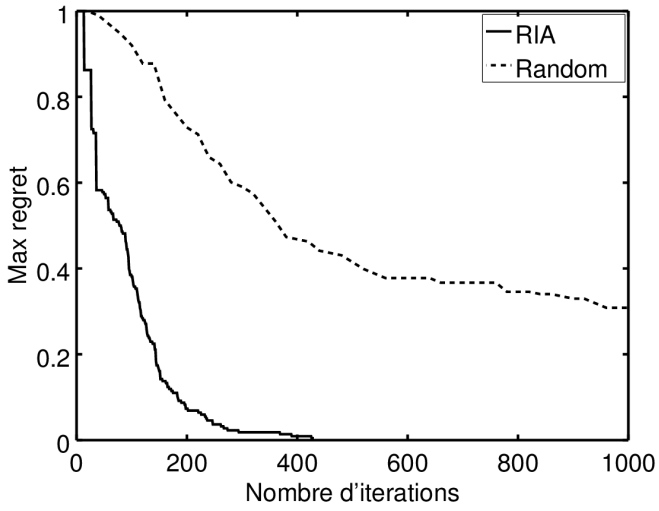


FIGURE 4.7 – MR du titulaire à chaque itération des procédures interactives ($q = 15$, $d = 0.5$, $n = 30$).

Dans la figure 4.7, nous observons que la valeur MR du titulaire réduit beaucoup plus rapidement avec RIA. Par exemple, après 200 itérations en moyenne, la valeur MR du titulaire est en dessous de 10% du regret initial avec RIA alors que celui-ci est toujours au dessus de 70% avec la procédure Random. Par ailleurs, avec RIA, ce regret réduit de manière significative dès les premières itérations, ce qui montre que notre procédure interactive est capable de déterminer des solutions de bonne qualité assez rapidement.

Nos dernières expériences ont pour objectif d'estimer les performances de RIA en terme de temps de calcul et de nombre de questions posées par agent. Les moyennes observées sur des instances avec $10, 20$ et 30 agents sont reportées dans la table 4.4.

	$q = 15$		$q = 18$		$q = 20$	
n	temps	questions	temps	questions	temps	questions
10	9.2	10.5	29.8	12.2	48.7	14.3
20	17.7	9.8	42.2	10.1	162.1	14.9
30	29.7	10.9	154.0	11.8	225.7	13.1

TABLE 4.4 – Performances de l'algorithme RIA (30 tests).

Dans la table 4.4, nous voyons que RIA est globalement très efficace. Par exemple, pour les problèmes avec 10 agents et 20 objets ($\approx$ 250000 solutions), RIA détermine un gagnant de l'élection en moins de 50 secondes avec moins de 15 questions d'approbation par agent. De plus, bien que le nombre de solutions réalisables croisse de manière exponentielle avec q le nombre d'objets, nous observons que le nombre moyen de questions par agent reste quant à lui plutôt stable. Ceci nous montre que la représentation compacte des préférences individuelles est globalement bien exploitée par RIA durant la recherche. Finalement, en comparant les tables 4.4 et 4.2, nous constatons que combiner recherche et élicitation incrémentale est une approche bien plus efficace en pratique que celle consistant à poser des questions avant de commencer la résolution. Par exemple, avec 20 agents, RIA parvient à déterminer un gagnant de l'élection avec moins de 15 questions par agent pour des problèmes avec environ 250000 solutions, alors que poser 20 questions par agent avant la résolution nous laisse avec encore 90 gagnants possibles sur de plus petits problèmes ($\approx$ 2000 solutions) tout en doublant les temps de calcul. Ceci illustre le fait que l'élicitation dirigée par la résolution permet de réduire à la fois les temps de calcul et le nombre de questions nécessaires pour pouvoir prendre une décision.

4.3 Conclusion du chapitre

Dans ce chapitre, nous nous sommes intéressés aux problèmes de décision collective en présence de préférences imprécisément connues. Dans le cadre de problèmes de vote avec préférences incomplètes, nous avons étudié le potentiel de l'élicitation incrémentale pour déterminer rapidement le gagnant de l'élection en limitant le nombre de questions posées aux votants.

Sur domaine non combinatoire, nous nous sommes concentrés sur des problèmes de vote où les alternatives sont évaluées sur différents critères à la fois. Dans ce contexte, en supposant que les préférences des agents sont représentables par une somme pondérée, nous avons proposé une procédure interactive fondée sur la notion de regret, permettant de déterminer un gagnant nécessaire pour la méthode de Borda. Cette méthode étend les travaux publiés dans [Lu and Boutilier, 2011b] au domaine multicritère. La principale difficulté provient de la gestion de la structure multicritère lors de l'optimisation des regrets. Dans la Section 4.1, nous avons introduit des programmes linéaires en nombres entiers pour résoudre ces problèmes d'optimisation, puis nous avons proposé des stratégies de génération de questions permettant de collecter des informations pertinentes sur les préférences des agents. Les expériences ont montré que notre méthode exploite relativement bien la structure multicritère des préférences, le gagnant de l'élection étant détecté avec un nombre de questions par agent assez faible.

Nous avons ensuite présenté une nouvelle approche pour le vote par approbation sur domaine combinatoire, illustrée sur le problème de sac à dos multi-agents. La première spécificité de notre approche est d'utiliser la représentation compacte des préférences individuelles par des utilités additives pour proposer des méthodes d'élicitation plus efficaces. En effet, dans ce cadre, apprendre que l'agent i approuve ou non une certaine alternative x n'est pas une information isolée. Celle-ci induit une contrainte sur l'espace des utilités admissibles pour l'agent i, ce qui permet de déduire implicitement des jugements

d'approbation sur d'autres alternatives du problème. Nous avons vu que cela contribue, en pratique, à une réduction importante du nombre de questions posées aux agents pour pouvoir déterminer les vainqueurs de l'élection. La seconde spécificité de notre approche est de combiner l'élicitation et la recherche. Cela permet en pratique d'éliciter les préférences des agents sur un domaine combinatoire, avec un double avantage en vue de la détermination des gagnants. D'une part, travailler avec des instanciations partielles dans l'arbre d'exploration facilite l'identification de questions pertinentes et réduit le nombre d'interactions nécessaires à la prise de décision. D'autre part, la recherche est plus rapidement focalisée sur les régions intéressantes de l'espace de solutions grâce aux questions posées à des moments décisifs de la résolution (ce qui permet de réduire drastiquement les temps de résolution). Cette approche nous a permis de résoudre des instances de grande taille pour lesquelles l'élicitation systématique des jugements d'approbation n'est pas une option envisageable.

Dans le chapitre suivant, nous nous intéressons aux problèmes de décision dans le risque en présence de préférences imprécisément spécifiées et proposons une nouvelle mise en œuvre de l'élicitation incrémentale sur domaine combinatoire.

Chapitre 5

Méthodes incrémentales pour la décision dans le risque sur domaine combinatoire

Ce chapitre, fondé sur nos travaux publiés dans [Benabbou and Perny, 2017], est consacré à la prise de décision dans le risque avec préférences incomplètes sur domaine combinatoire. Plus précisément, nous nous intéressons aux problèmes de décision séquentielle dans le risque dont les stratégies possibles sont décrites par un arbre de décision. Dans ce contexte, il n'est pas envisageable d'énumérer toutes les stratégies possibles dans le but d'identifier celle que les agents décisionnels préfèrent, car le nombre de stratégies dans un arbre de décision est exponentiel en la taille de l'arbre. Pour atteindre ce but plus efficacement, nous proposons de considérer une approche mêlant la résolution de l'arbre de décision et l'élicitation incrémentale des préférences. Plus précisément, nous introduisons ici un algorithme interactif par induction arrière permettant de construire en temps polynomial une stratégie (presque) optimale au sens de l'utilité sociale espérée, en posant un nombre polynomial de questions dans le pire cas.

5.1 Introduction et formalisation du problème

Dans les chapitres précédents, nous avons considéré des problèmes de décision dans lesquels les différents agents impliqués avaient une unique décision à prendre. Cependant, très souvent, il ne s'agit pas de prendre une unique décision mais de déterminer la meilleure séquence de décisions, étalées dans le temps, en fonction des différentes réalisations d'événements de la nature ; une telle séquence de décision est communément appelée *stratégie* dans la littérature. Lorsque les probabilités sur les événements possibles sont connus, on parle de problèmes de *décision séquentielle dans le risque*. À titre illustratif, un exemple de problème de décision séquentielle dans le risque est donné ci-dessous :

Exemple 32. *Un couple envisage d'investir* 100 *euros pour commencer à cultiver ses fruits et légumes dans son jardin domestique, dans l'optique de faire des économies sur long terme (estimées à* 1000 *euros). Ces deux personnes sont au courant que, parmi leurs voisins qui ont tenté de faire leur propre potager,* 30% *n'y sont pas parvenus à cause de la structure et de la fertilité de leur sol. Avant de prendre sa décision, le couple a la possibilité de payer* 200 *euros pour qu'un spécialiste vienne tester la qualité de son sol. Si le test s'avère positif, alors la probabilité que ce couple rencontre des difficultés passe à* 10% *(au lieu de* 30% *sans information). En revanche, si le résultat est négatif, cette probabilité monte à* 80%*. Par ailleurs,* 40% *des tests effectués par ce spécialiste dans la région se sont révélés positifs. En prenant en compte toutes ces informations, le couple doit décider de faire appel ou non au spécialiste, puis décider de cultiver ou non son jardin.*

Les problèmes de décision séquentielle dans le risque sont relativement fréquents dans la vie de tous les jours. Nous pouvons par exemple penser à une entreprise cherchant à définir sa stratégie marketing sur plusieurs années, ou encore à des médecins luttant contre une maladie à l'aide de divers traitements à conséquences non certaines. Afin de déterminer le meilleur plan d'action possible, les systèmes de planification multi-agents ont besoin de modèles informatiques conçus pour la prise de décision collective en situation d'incertitude. En effet, ces derniers permettent non seulement de contrôler le choix des décisions à prendre dans les différents états possibles, mais aussi de tenir compte des situations qui peuvent se présentées dans le futur suite à leurs actions (e.g., [Boutilier et al., 1999, Guestrin et al., 2001, Goldman and Zilberstein, 2008, Wooldridge, 2009, Wu et al., 2011]). Dans ce chapitre, nous nous intéressons plus particulièrement aux problèmes de décision séquentielle dans le risque représentés sous la forme d'arbres de décision, qui sont des outils formels offrant la possibilité de modéliser de manière simple une très grande partie des problèmes de décision séquentielle rencontrés en pratique.

5.1.1 Les arbres de décision comme modèle graphique

Dans la littérature, il existe différents modèles graphiques permettant de représenter un problème de décision séquentielle dans le risque, comme les diagrammes d'influence (e.g., [Howard and Matheson, 1984]) ou encore les processus décisionnels de Markov (e.g., [Bellman, 1957]). Dans ce chapitre, nous nous intéressons aux arbres de décision (e.g., [Raiffa, 1968]), car cette modélisation offre une représentation simple et explicite de nombreux problèmes de décision séquentielle dans le risque. Plus précisément, ce

modèle graphique permet de représenter toute situation de décision dans le risque se décomposant en un nombre fini de décisions, conditionnées par un nombre fini d'évènements. Un arbre de décision est ce que l'on appelle une *arborescence*, c'est-à-dire un graphe orienté acyclique dans lequel on distingue un nœud racine (sans parent), les autres nœuds ayant exactement un parent. Un arbre de décision $\mathcal{A} = (\mathcal{V}, \mathcal{E})$ est constitué d'un ensemble fini de nœuds $\mathcal{V}$ et d'un ensemble fini d'arcs $\mathcal{E}$, l'ensemble $\mathcal{V}$ se partitionnant en trois sous-ensembles :

- $\mathcal{V}_D$: chaque nœud de cet ensemble représente un état dans lequel les agents peuvent être amenés à prendre une décision. Chaque arc sortant d'un tel nœud modélise une décision possible dans cet état. Ces nœuds sont appelés *nœuds de décision* et sont représentés graphiquement par un carré.
- $\mathcal{V}_C$: chaque nœud de cet ensemble représente un état où ce n'est pas aux agents de prendre une décision mais la nature. Celle-ci intervient dans le processus de décision sous la forme d'une variable aléatoire. Plus précisément, les arcs sortant d'un tel nœud correspondent aux événements possibles et sont étiquetés par leur probabilité sachant toutes les décisions qui ont été prises précédemment et tous les événements qui se sont déjà produits. Ces nœuds sont appelés *nœuds de chance* et sont représentés graphiquement par des ronds.
- $\mathcal{V}_C$: cet ensemble correspond aux feuilles de l'arbre et représente l'ensemble des conséquences possibles pour le problème de décision séquentielle considéré.

Afin de rendre cette définition un peu plus concrète, nous donnons ci-après l'arbre de décision associé au problème de décision séquentielle présenté dans l'exemple 32 :

Exemple 33. *Nous souhaitons construire l'arbre de décision correspondant au problème de décision séquentielle dans le risque décrit dans l'exemple 32. Dans cette situation, le couple doit dans un premier temps décider de faire appel à un spécialiste (décision s) ou non (décision $\bar{s}$), puis décider de cultiver (décision c) ou non (décision $\bar{c}$). Notons R_{pot} la variable aléatoire représentant le résultat obtenu après une tentative de création de potager ; les éventualités possibles sont un échec (noté E) ou une réussite (noté R). Par ailleurs, notons R_{test} la variable aléatoire représentant la conclusion du spécialiste ; les éventualités possibles sont notées P et N pour un résultat positif et négatif respectivement. Rappelons que nous avons les probabilités a priori suivantes :*

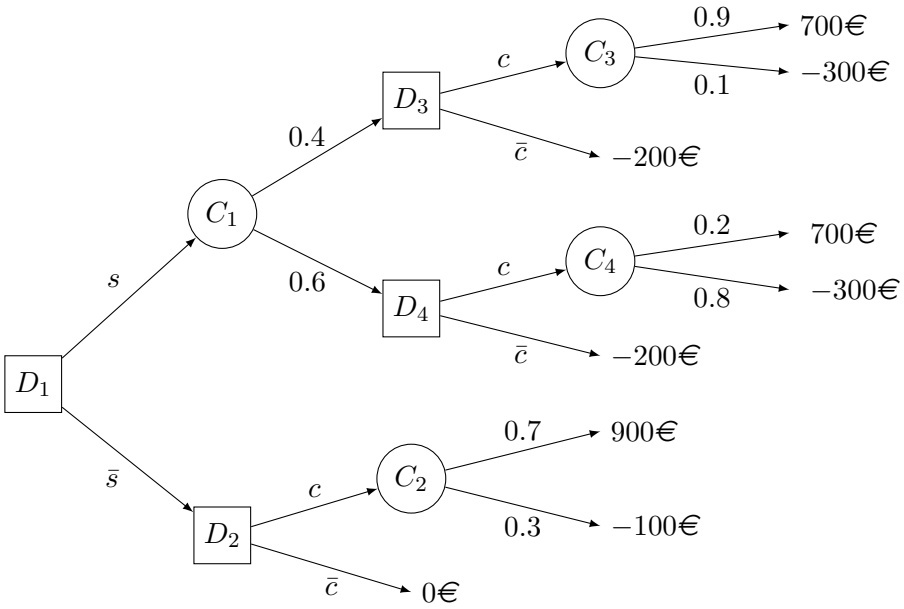
$$P(R_{pot} = R) = 0.7,\ P(R_{pot} = E) = 0.3,\ P(R_{test} = P) = 0.4,\ P(R_{test} = N) = 0.6$$

Comme le couple doit décider de cultiver ou non son jardin après avoir décider de faire appel ou non au spécialiste (décision séquentielle), nous avons aussi besoin de connaître la distribution de probabilité de R_{pot} sachant R_{test}. Celle-ci est rappelée ci-dessous :

$$P(R_{pot} = R \mid R_{test} = P) = 0.9,\ P(R_{pot} = E \mid R_{test} = P) = 0.1,$$
$$P(R_{pot} = R \mid R_{test} = N) = 0.2,\ P(R_{pot} = E \mid R_{test} = N) = 0.8.$$

Enfin, pour pouvoir construire l'arbre de décision, il est nécessaire de connaître les conséquences qui

découlent de leurs actions. Rappelons que tenter de mettre en place un potager coûte 100 *euros, que sa réussite dégage un gain de* 1000 *euros et que le spécialiste demande* 200 *euros. Ainsi, l'arbre de décision représentant ce problème de décision séquentielle dans le risque est le suivant :*

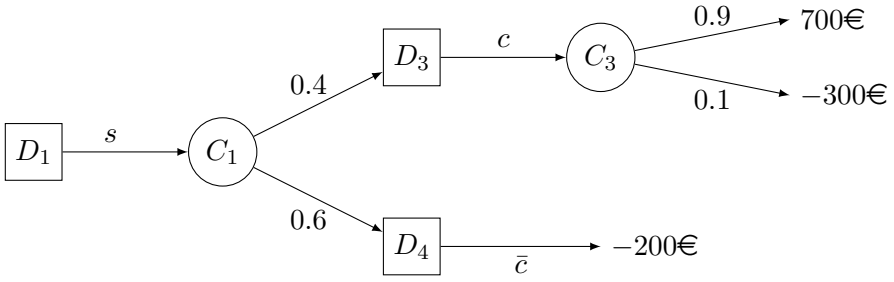


Un arbre de décision est généralement représenté de gauche à droite pour rappeler l'aspect séquentiel du processus décisionnel. Chaque chemin de l'arbre correspond à un déroulement possible des événements. Décrivons par exemple le chemin du haut : le nœud D_1 *représente l'état où le couple doit prendre sa première décision. À ce nœud, il choisit de payer le spécialiste (décision* s*), ce qui l'amène au niveau du nœud chance* C_1 *modélisant le résultat du test réalisé par le spécialiste. Comme le test s'avère positif, alors le couple se retrouve ensuite au niveau du nœud de décision* D_3*. À ce niveau, il s'agit pour le couple de prendre une décision concernant la création du potager, sachant que le test a réussi. Le couple décide alors de lancer la création de son potager, ce qui le conduit au nœud de chance* C_3 *modélisant sa réussite ou son échec. Finalement, dans ce déroulement des événements, le couple parvient à avoir son potager et dégage un gain de* 700 *(*$= 1000 - 200 - 100$*) euros.*

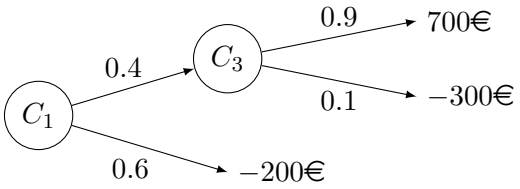
Dans un arbre de décision, une stratégie consiste à sélectionner une décision possible par nœud de décision de l'arbre. Autrement dit, chaque ensemble d'arcs contenant exactement un arc sortant par nœud de décision permet de caractériser une stratégie possible du problème. À titre d'exemple, l'ensemble d'arcs $\{(D_1, C_1), (D_2, 0€), (D_3, C_3), (D_4, -200€)\}$ définit une stratégie possible pour l'arbre de l'exemple 33. Remarquons que l'information $(D_2, 0€)$ ne semble pas pertinente pour cette stratégie, car le nœud de décision D_2 n'est plus accessible dès lors que la décision (D_1, C_1) a été prise. Plus généralement, nous pouvons résumer une stratégie en enlevant tous les arcs associés à des nœuds de décision qui ne peuvent plus être atteints au vu des décisions qui les précèdent dans le temps ; en particulier, la stratégie $\{(D_1, C_1), (D_2, 0€), (D_3, C_3), (D_4, -200€)\}$ se résume à l'ensemble $\{(D_1, C_1), (D_3, C_3), (D_4, -200€)\}$.

Dans un arbre de décision, nous appellerons *sous-arbre* tout sous-graphe défini par un nœud de décision donné et l'ensemble de ses descendants ; la restriction d'une stratégie à un sous-arbre sera quant à elle appelée *sous-stratégie*. Par ailleurs, un nœud de décision sera dit *accessible* en suivant une (sous-)stratégie si et seulement s'il existe une réalisation des événements telle que cette (sous-)stratégie mène à un moment donné à ce nœud de décision.

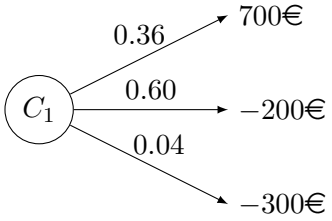
À chaque stratégie possible, nous pouvons associer une loterie simple représentant la distribution de probabilité sur les conséquences obtenues en suivant la stratégie. Pour construire la loterie simple associée à une stratégie, nous commençons par élaguer tous les sous-arbres correspondant aux décisions ne faisant pas partie de la stratégie. Par exemple, pour la stratégie $s_1 = \{(D_1, C_1), (D_3, C_3), (D_4, -200€)\}$, nous obtenons le sous-graphe suivant :



Il s'agit ensuite de faire disparaître tous les nœuds de décision (ainsi que leur unique arc sortant) puisque ces derniers n'ont plus aucun impact sur la distribution de probabilité. Pour la stratégie s_1, nous obtenons alors la loterie composée suivante :



En supposant que les préférences des agents vérifient l'axiome de réduction des loteries composées (cf. Section 1.3.2), la loterie composée ainsi construite est équivalente à la loterie simple qui associe les mêmes conséquences avec les mêmes probabilités. La loterie simple correspondant à la stratégie s_1 est :



Avec l'axiome de réduction des loteries composées, comparer des stratégies dans un arbre de décision revient à comparer les loteries simples qui leur sont associées. Dans la suite, nous noterons $\mathcal{L}_{\mathcal{A}}$ l'ensemble des loteries simples associées à une stratégie possible dans l'arbre de décision $\mathcal{A}$.

5.1.2 Modélisation des préférences

Considérons un problème de décision collective où un ensemble $N = \{1, \ldots, n\}$ d'agents se retrouve face à un problème de décision séquentielle dans le risque modélisé sous la forme d'un arbre de décision noté $\mathcal{A} = (\mathcal{V}, \mathcal{E})$. Nous faisons ici l'hypothèse que l'ensemble $\mathcal{V}_C$ des conséquences possibles sont des valeurs réelles à maximiser (e.g., bénéfices, performances, durées). Par ailleurs, nous supposons que les différents agents impliqués dans la prise de décision ont leur propre système de valeurs mais qu'ils sont prêts à coopérer pour maximiser le bien-être social. Dans ce contexte, notre objectif est d'étudier des approches centralisées pour la prise de décision collective par élicitation incrémentale des préférences. Les principales questions à traiter concernent alors la gestion de préférences incomplètes au sein du processus décisionnel ainsi que la génération de questions informatives pour identifier rapidement la meilleure stratégie pour notre groupe d'agents.

Nous adoptons ici la théorie de l'utilité espérée aussi bien au niveau individuel que collectif (e.g., [von Neumann and Morgenstern, 1947]). Plus précisément, au niveau individuel, nous supposons que tous les agents cherchent à maximiser leur utilité espérée et que leurs préférences sont compatibles avec les postulats de la théorie de von Neuman et Morgenstern (vNM) ; en particulier, nous faisons l'hypothèse que les préférences des agents vérifient l'axiome de réduction des loteries composées. Au niveau collectif, nous adoptons ici le point de vue d'un coordinateur en charge du bien-être social, agissant tel un *observateur impartial* guidé par des notions d'intérêt collectif et de justice impartiale, comme considéré dans le cadre du théorème d'Harsanyi (e.g., [Harsanyi, 1978]). Le théorème d'Harsanyi peut se formuler comme suit : si les préférences des agents et de l'observateur impartial sont compatibles avec la théorie de vNM et que les préférences de l'observateur impartial vérifient le principe de Pareto fort [1], alors la fonction d'utilité de l'observateur impartial est une agrégation linéaire des utilités individuelles avec des poids strictement positifs. Par ailleurs, lorsque les comparaisons interpersonnelles des utilités sont autorisées et que les individus doivent être traités de manière équivalente par le système, John C. Harsanyi suggère d'utiliser des poids égaux et de définir la fonction de bien-être social par $w = \sum_{i=1}^{n} u_i$, où les fonctions u_i représentent les utilités individuelles des n agents. Cette justification de la règle utilitaire pure a été largement discutée et critiquée dans la litérature (e.g., [Sen, 1986, Weymark, 1991]), mais le résultat initial a été utilement revisité et reformulé dans différents contextes plus généraux (e.g., [Hammond, 1992, Mongin, 1994, Blackorby et al., 2008, Fleurbaey and Mongin, 2016]). Par exemple, le formalisme multi-profils considéré dans [Mongin, 1994] étudie les fonctions de bien-être social admettant plusieurs profils d'utilités $(u_1, \ldots, u_n)$ en entrée. Dans ce cadre, une caractérisation équivalente au théorème d'Harsanyi a été proposée, permettant de déterminer les poids indépendamment des utilités données, et d'obtenir des poids tous égaux en introduisant une condition d'anonymat (impliquant la symétrie de l'opérateur d'agrégation). Ce résultat ainsi que d'autres mentionnés ci-dessus améliorent la justification initiale du critère utilitaire pure pour la prise de décision collective avec des utilités de vNM.

1. Le principe de Pareto fort nécessite que, si tous les agents préfèrent la situation sociale A à la situation sociale B, avec au moins un de ces agents qui préfère strictement A à B, alors la société (ici l'observateur impartial) doit préférer strictement A à B.

En cohérence avec le cadre décrit ci-dessus, nous supposons que les préférences de chaque agent $i \in N$ sont représentables par le modèle EU (pour *Expected Utility* en anglais) définie par :

$$EU(\ell, u_i) = \sum_{k=1}^{m} p_k u_i(x_k)$$

pour toute loterie $\ell = (x_1, p_1; \ldots; x_m, p_m)$, où $u_i : \mathbb{R} \to \mathbb{R}$ est la fonction d'utilité de vNM de l'agent i. Plus précisément, la loterie $\ell_1 \in \mathcal{L}_\mathcal{A}$ est meilleure que la loterie $\ell_2 \in \mathcal{L}_\mathcal{A}$ pour l'agent i si et seulement si $EU(\ell_1, u_i) \geq EU(\ell_2, u_i)$. Par ailleurs, étant donné le profil $u = (u_1, \ldots, u_n)$, les préférences sociales sont caractérisées par l'utilité de vNM $w = \sum_{i=1}^{n} u_i$. Les préférences sociales sont donc représentables par l'utilité sociale espérée EW (pour *Expected social Welfare* en anglais) définie par :

$$EW(\ell, u) = EU(\ell, w) = \sum_{i=1}^{n} EU(\ell, u_i)$$

pour toute loterie $\ell \in \mathcal{L}_\mathcal{A}$. La loterie $\ell_1 \in \mathcal{L}_\mathcal{A}$ est meilleure que la loterie $\ell_2 \in \mathcal{L}_\mathcal{A}$ pour notre groupe d'agents si et seulement si $EW(\ell_1, w) \geq EW(\ell_2, w)$. Par conséquent, dans notre contexte, tout élément de l'ensemble $\arg\max_{\ell \in \mathcal{L}_\mathcal{A}} EW(\ell, w)$ correspond à une stratégie optimale dans l'arbre de décision $\mathcal{A}$.

Comme $EW(\ell, u) = EU(\ell, w)$ par définition, alors les méthodes de résolution conçues pour le modèle EU peuvent être utilisées dans notre contexte multi-agents. En particulier, lorsque l'utilité de vNM $w = \sum_{i \in N} u_i$ est connue précisément, une stratégie optimale peut être construite en remontant l'arbre de décision à partir des feuilles et en réalisant, au niveau de chaque nœud de décision, une sélection fondée sur le critère $EU(\cdot, w)$. En effet, il est bien connu qu'un tel algorithme, dit par *induction arrière*, permet de déterminer (en temps polynomial) une stratégie EU-optimale dans un arbre de décision, car toute stratégie EU-optimale vérifie le principe d'optimalité suivant : toute sous-stratégie d'une stratégie optimale est elle aussi optimale parmi les sous-stratégies possibles dans le même sous-arbre (e.g., [Hammond, 1988]). Par exemple, si la stratégie $\{(D_1, C_1), (D_3, C_3), (D_4, -200€)\}$ est EU-optimale dans le problème de l'exemple 33, alors la sous-stratégie définie par $\{(D_4, -200€)\}$ est forcément EU-optimale dans le sous-arbre enraciné en D_4 et la sous-stratégie définie par $\{(D_3, C_3)\}$ est quant à elle nécessairement EU-optimale dans le sous-arbre enraciné en D_3.

Cependant, en pratique, il est difficile pour les agents de fournir directement les utilités permettant de modéliser précisément leurs préférences. À la place, ces utilités doivent être élicitées au préalable en collectant des informations sur les préférences des agents. Néanmoins, l'élicitation totale des fonctions d'utilité de vNM nécessite en pratique de poser de nombreuses questions aux agents pour être réalisée de manière précise. En pratique, nous pouvons uniquement observer quelques informations du type "je préfère ℓ_1 à ℓ_2", où ℓ_1 et ℓ_2 sont deux loteries relativement simples. Dans ces situations, nous devons considérer, pour chaque agent $i \in N$, l'ensemble U_i de toutes les fonctions croissantes $u_i : \mathbb{R} \to \mathbb{R}$ compatibles avec l'ensemble $\mathcal{P}_i$ des données de préférences disponibles concernant l'agent i. Plus précisément, toute paire $(\ell_1, \ell_2) \in \mathcal{P}_i$, représentant le fait que l'agent i préfère ℓ_1 à ℓ_2, induit la contrainte $EU(\ell_1, u_i) \geq EU(\ell_2, u_i)$

sur l'ensemble des fonctions u_i admissibles. Dans ce chapitre, nous supposons que les éventuelles informations disponibles sur les préférences des agents nous permettent de réduire l'ensemble $U = U_1 \times \ldots \times U_n$ des profils d'utilités $u = (u_1, \ldots, u_n)$ admissibles, mais que celles-ci ne sont pas suffisantes pour pouvoir construire précisément les fonctions d'utilité de vNM des agents. Dans ce contexte, nous nous intéressons à la prise de décision collective par élicitation incrémentale des préférences.

À titre préliminaire, nous allons commencer par étudier le problème de détermination de toutes les loteries *potentiellement optimales* dans l'arbre de décision, c'est-à-dire toutes les loteries $\ell \in \mathcal{L}_{\mathcal{A}}$ qui maximisent $EW(\ell, u)$ pour au moins un profil d'utilités $u = (u_1, \ldots, u_n)$ admissible. Puisque les loteries possibles dans un arbre de décision sont très nombreuses et définies de manière implicite, nous proposons un algorithme par induction arrière pour la détermination efficace des loteries potentiellement optimales. Soulignons le fait que ce problème s'écarte des études précédentes dans les arbres de décision qui considèrent le problème de détermination des stratégies potentiellement optimales au sens de préférences partielles induites par des probabilités imprécises (e.g., [Kikuti et al., 2011]). Dans notre cas, l'arbre de décision est complètement spécifié et les probabilités sur les événements possibles sont connus précisément ; l'indétermination provient uniquement de l'imprécision sur les utilités de vNM. Dans un second temps, nous proposerons une procédure de recherche par élicitation incrémentale permettant de déterminer une loterie nécessairement optimale en temps polynomial avec un nombre polynomial de questions dans le pire cas.

5.2 Calcul des solutions potentiellement optimales au sens de l'utilité sociale espérée

Étant donné un ensemble de profils admissibles $U = U_1 \times \ldots \times U_n$ et un ensemble de loteries L quelconque, l'ensemble $\mathrm{PO}_U(L)$ des loteries potentiellement optimales est défini par :

$$\mathrm{PO}_U(L) = \bigcup_{u \in U} \arg\max_{\ell \in L} EW(\ell, u) \tag{5.1}$$

Dans cette sous-section, nous nous intéressons au calcul efficace de l'ensemble $\mathrm{PO}_U(\mathcal{L}_{\mathcal{A}})$, c'est-à-dire l'ensemble des loteries potentiellement optimales parmi celles associées à des stratégies dans l'arbre $\mathcal{A}$. Cet ensemble de loteries peut être construit en utilisant la relation de dominance suivante :

Définition 55 ($\succ_U$-dominance)**.** *Pour tout ensemble de loteries L et toute loterie ℓ_0 :*

$$L \succ_U \ell_0 \Leftrightarrow \forall u \in U, \exists \ell \in L,\ EW(\ell, u) > EW(\ell_0, u)$$

Cette relation de dominance vérifie en effet la propriété intéressante suivante :

Proposition 39. *Pour tout ensemble de loteries L et toute loterie $\ell_0 \in L$:*

$$\ell_0 \in PO_U(L) \Leftrightarrow \forall L' \subseteq L, \neg(L' \succ_U \ell_0)$$

où $\neg$ représente l'opérateur de négation logique.

Démonstration. ($\Rightarrow$) Soit $\ell_0 \in \mathrm{PO}_U(L)$. Supposons qu'il existe un ensemble de loteries $L' \subseteq L$ tel que nous ayons $L' \succ_U \ell_0$. Par définition de la $\succ_U$-dominance (cf. définition 55), pour tout profil $u \in U$, il existe une loterie $\ell \in L'$ telle que $EW(\ell, u) > EW(\ell_0, u)$. Comme l'inclusion $L' \subseteq L$ est vraie par hypothèse, alors pour tout profil admissible $u \in U$, nous avons $\ell_0 \notin \arg\max_{\ell \in L} EW(\ell, u)$. En utilisant l'équation (5.1) définissant $\mathrm{PO}_U(L)$, nous en déduisons que la loterie ℓ_0 ne peut pas appartenir à l'ensemble $\mathrm{PO}_U(L)$, ce qui contredit notre hypothèse de départ.

($\Leftarrow$) Soit $\ell_0 \in L$ une loterie telle que $\neg(L' \succ_U \ell_0)$ pour tout ensemble de loteries $L' \subseteq L$. En particulier, nous avons $\neg(L \succ_U \ell_0)$. Par définition de la $\succ_U$-dominance, nous en déduisons qu'il existe un profil admissible $u \in U$ tel que $EW(\ell, u) \leq EW(\ell_0, u)$ pour toute loterie $\ell \in L$. Par conséquent, nous avons forcément $\ell_0 \in \mathrm{PO}_U(L)$ par définition de $\mathrm{PO}_U(L)$ (cf. équation (5.1)). □

Cette proposition nous donne une autre caractérisation de l'ensemble des loteries potentiellement optimales dans l'arbre. Plus précisément, $\mathrm{PO}_U(\mathcal{L}_\mathcal{A})$ est l'ensemble des loteries non $\succ_U$-dominées dans $\mathcal{L}_\mathcal{A}$. Signalons que, dans la littérature, il existe des travaux sur la détermination des solutions non dominées sur domaine combinatoire, pour des relations binaires quasi-transitives représentant des préférences (e.g., [Perny and Spanjaard, 2005]). Nos travaux se distinguent de ces derniers en considérant une relation de dominance qui est fondée sur des préférences imprécises et qui utilise des ensembles de loteries pour dominer une seule loterie.

Pour déterminer l'ensemble $\mathrm{PO}_U(\mathcal{L}_\mathcal{A})$, nous introduisons maintenant un algorithme par induction arrière avec des sélections fondées sur la $\succ_U$-dominance. Pour simplifier la présentation, nous supposons sans perte de généralité que la racine de l'arbre $\mathcal{A}$ est un nœud de décision et que les nœuds de décision sont indexés suivant un ordre topologique : s'il existe un chemin dans $\mathcal{A}$ partant du nœud $D_k \in \mathcal{V}_D$ et arrivant au nœud $D_l \in \mathcal{V}_D$, alors nous avons forcément $k < l$. En commençant par le dernier nœud de décision dans l'ordre topologique, il s'agit de faire remonter dans l'arbre uniquement les loteries non $\succ_U$-dominées en chaque nœud de décision, jusqu'à atteindre la racine de l'arbre. Cette idée est implémentée dans l'algorithme 9. Dans cet algorithme, chaque nœud de décision D_k garde un ensemble L_k de loteries associées à des sous-stratégies du sous-arbre enraciné en D_k. Cet ensemble, stocké dans l'attribut nommé *loteries*, est calculé en examinant tous les enfants A du nœud D_k dans l'arbre $\mathcal{A}$:

- Si A est un nœud de décision, alors toutes les loteries qui ont été stockées dans A (durant une itération antérieure) sont ajoutées à l'ensemble L_k.
- Si le nœud A est une feuille de l'arbre, alors la loterie $(A, 1)$ associant la conséquence A avec une probabilité de 1 est ajoutée à l'ensemble L_k.
- Si A est un nœud chance, alors toutes les combinaisons de loteries stockées dans les descendants de A doivent être insérées dans l'ensemble L_k. Ces combinaisons sont calculées à l'aide de l'algorithme récursif nommé Combinaison, à partir de l'appel initial Combinaison$(A, \mathcal{A})$, où $p(A, A_l)$ représente la probabilité associée à l'arc (A, A_l) de l'arbre $\mathcal{A}$.

Finalement, seuls les éléments non $\succ_U$-dominés dans l'ensemble L_k sont stockés dans l'attribut *loteries* du nœud de décision D_k.

Algorithm 9: Induction arrière

Input: $\mathcal{A} = (\mathcal{V}, \mathcal{E})$: un arbre de décision ; $U = U_1 \times \ldots \times U_n$: un ensemble de profils admissibles
Output: L'ensemble $\mathrm{PO}_U(\mathcal{L}_\mathcal{A})$

for $k = |\mathcal{V}_D|, \ldots, 1$ **do**
  $L_k \leftarrow \emptyset$
  forall $(D_k, A) \in \mathcal{E}$ **do**
    if $A \in \mathcal{V}_D$ **then**
      $L_k \leftarrow L_k \cup A.loteries$
    else if $A \in \mathcal{V}_C$ **then**
      $L_k \leftarrow L_k \cup$ Combinaison$(A, \mathcal{A})$
    else
      `/* cas où` A `est une feuille de l'arbre */`
      $\ell \leftarrow (A, 1)$
      $L_k \leftarrow L_k \cup \{\ell\}$
    end
  end
  $D_k.loteries \leftarrow \{\ell \in L_k : \forall L \subseteq L_k, \neg(L \succ_U \ell)\}$
end
return $D_1.loteries$

Algorithm 10: Combinaison

Input: A : un nœud chance ; $\mathcal{A}$: un arbre de décision contenant A
Output: L'ensemble des loteries obtenu en combinant toutes les loteries stockées dans les nœuds enfants du nœud A.

Soient $A_1, \ldots A_m$ les m enfants du nœud A dans $\mathcal{A}$
for $l = 1, \ldots, m$ **do**
  if $A_l \in \mathcal{V}_D$ **then**
    $L'_l \leftarrow A_l.loteries$
  else if $A_l \in \mathcal{V}_C$ **then**
    $L'_l \leftarrow$ Combinaison(A_l)
  else
    `/* cas où` A_l `est une feuille de l'arbre */`
    $\ell \leftarrow (A_l, 1)$
    $L'_l \leftarrow \{\ell\}$
  end
end
$L \leftarrow \emptyset$
forall $(\ell_1, \ldots, \ell_m) \in L'_1 \times \ldots \times L'_m$ **do**
  $L \leftarrow L \cup \{\sum_{l=1}^{m} p(A, A_l)\ell_l\}$
end
return L

Les propositions suivantes vont nous permettre de montrer que notre algorithme de recherche par induction arrière est valide.

Proposition 40 (Indépendance)**.** *Pour tout ensemble de loteries L et toute loterie ℓ_0 :*

$$L \succ_U \ell_0 \Rightarrow \{\lambda\ell + (1-\lambda)\ell' : \ell \in L\} \succ_U \lambda\ell_0 + (1-\lambda)\ell'$$

pour toute loterie ℓ' et tout nombre réel $\lambda \in]0,1[$.

Démonstration. Notons $\succ_u$ la relation de préférence sociale définie par $\ell_1 \succ_u \ell_2$ si et seulement si $EW(\ell, u) > EW(\ell_2, u)$. Soient L et ℓ_0 tels que $L \succ_U \ell_0$. Considérons par ailleurs une loterie ℓ' et une valeur réelle $\lambda \in]0,1[$ quelconques. Comme nous avons $L \succ_U \ell_0$, alors pour tout profil $u \in U$, il existe une loterie $\ell \in L$ telle que $\ell \succ_u \ell_0$ (par définition de la $\succ_U$-dominance). Puisque que la relation de préférence sociale $\succ_u$ respecte la théorie de vNM, alors nous avons $\lambda\ell + (1-\lambda)\ell' \succ_u \lambda\ell_0 + (1-\lambda)\ell'$ d'après l'axiome d'indépendance de la théorie de vNM (cf. Section 1.3.2). Par conséquent, nous avons $EW(\lambda\ell + (1-\lambda)\ell', u) > EW(\lambda\ell_0 + (1-\lambda)\ell', u)$ par définition de la relation $\succ_u$. En utilisant la définition de la $\succ_U$-dominance (cf. définition 55), nous en déduisons $\{\lambda\ell + (1-\lambda)\ell' : \ell \in L\} \succ_U \lambda\ell_0 + (1-\lambda)\ell'$. □

Notons $\mathcal{L}_k$ l'ensemble des loteries associées à une sous-stratégie enracinée en D_k. Nous avons alors le résultat suivant :

Proposition 41. *Soient deux nœuds $D_k, D_l \in \mathcal{V}_D$ tels que D_l est un descendant de D_k dans $\mathcal{A}$. Soit ℓ_k la loterie associée à une sous-stratégie s enracinée en D_k. Soit ℓ_l la loterie associée à la sous-stratégie de s enracinée en D_l. Si ℓ_l est $\succ_U$-dominée dans $\mathcal{L}_l$ alors ℓ_k est $\succ_U$-dominée dans $\mathcal{L}_k$.*

Démonstration. Comme ℓ_l est $\succ_U$-dominée dans $\mathcal{L}_l$, alors il existe $L \subseteq \mathcal{L}_l$ tel que $L \succ_U \ell_l$. Montrons alors que la loterie ℓ_k est forcément $\succ_U$-dominée dans $\mathcal{L}_k$. Puisque le nœud D_l est un descendant du nœud D_k dans l'arbre $\mathcal{A}$, alors il existe une unique chemin partant du nœud D_k et menant au nœud D_l. Soient $(A_1, \ldots, A_m)$ la séquence de nœuds rencontrés le long de ce chemin, en particulier, nous avons $A_1 = D_k$ et $A_m = D_l$. Deux cas peuvent alors se produire :

- *Cas où $A_j \in \mathcal{V}_D$ pour tout $j \in \{1, \ldots, m\}$:* dans ce cas, nous avons nécessairement $\ell_k = \ell_l$ ainsi que $L \subseteq \mathcal{L}_k$. Par conséquent, en utilisant les hypothèses, nous en déduisons $L \succ_U \ell_k$ et donc la loterie ℓ_k est $\succ_U$-dominée dans $\mathcal{L}_k$.
- *Cas où $A_j \in \mathcal{V}_C$ pour au moins un indice $j \in \{1, \ldots, m\}$:* dans ce cas, la loterie ℓ_k est forcément de la forme $\ell_k = \lambda\ell_l + (1-\lambda)\ell'$ où :

  $$\lambda = \prod_{j : A_j \in \mathcal{V}_C} p(A_j, A_{j+1})$$

  $p(A_j, A_{j+1})$ représentant la probabilité étiquetée sur l'arc (A_j, A_{j+1}). En utilisant la propriété d'indépendance (cf. proposition 40), nous déduisons ensuite $L' \succ_U \ell_k$ de notre hypothèse $L \succ_U \ell_l$, où $L' = \{\lambda\ell + (1-\lambda)\ell' : \ell \in L\}$. Comme $L' \subseteq \mathcal{L}_k$, alors nous en déduisons que la loterie ℓ_k est $\succ_U$-dominée dans $\mathcal{L}_k$.

□

Proposition 42 (Transitivité)**.** *Pour tous ensembles de loteries L, L' et pour toutes loteries ℓ_0, ℓ_1 :*

$$L' \succ_U \ell_1 \text{ et } L \cup \{\ell_1\} \succ_U \ell_0 \Rightarrow L \cup L' \succ_U \ell_0$$

Démonstration. Soient L, L', ℓ_0, ℓ_1 tels que $L' \succ_U \ell_1$ et $L \cup \{\ell_1\} \succ_U \ell_0$. Nous voulons montrer que nous avons forcément $L \cup L' \succ_U \ell_0$. Pour y parvenir, il suffit de prouver que pour tout profil admissible $u \in U$, il existe une loterie $\ell \in L \cup L'$ telle que $EW(\ell, u) > EW(\ell_0, u)$ (par définition de la $\succ_U$-dominance). Soit $u \in U$ un profil d'utilités admissible quelconque. Puisque $L \cup \{\ell_1\} \succ_U \ell_0$ par hypothèse, alors il existe une loterie $\ell \in L \cup \{\ell_1\}$ telle que $EW(\ell, u) > EW(\ell_0, u)$ (par définition de la $\succ_U$-dominance). Deux cas sont alors possibles : soit $\ell \in L$ soit $\ell = \ell_1$. Dans le premier cas, le résultat est directement établi puisque la loterie ℓ appartient trivialement à l'ensemble $L \cup L'$. Supposons maintenant que $\ell = \ell_1$. Dans ce cas, puisque $L' \succ_U \ell_1$ par hypothèse, alors il existe une loterie $\ell' \in L'$ telle que $EW(\ell', u) > EW(\ell_1, u)$ (encore par définition de la $\succ_U$-dominance). Par conséquent, nous obtenons $EW(\ell', u) > EW(\ell_1, u) > EW(\ell_0, u)$ par transitivité des inégalités. Comme $\ell' \in L'$ par définition, alors nous avons à la fois $\ell' \in L \cup L'$ et $EW(\ell', u) > EW(\ell_0, u)$, ce qui nous permet de conclure la preuve. □

Proposition 43. *Pour tout ensemble de loteries L :*

$$\forall \ell_0 \in L,\ \ell_0 \notin PO_U(L) \Rightarrow PO_U(L) \succ_U \ell_0$$

Démonstration. Soit $\ell_0 \notin PO_U(L)$. Supposons que $\neg(PO_U(L) \succ_U \ell_0)$. Dans ce cas, il existe un profil d'utilités admissible $u \in U$ tel que $EW(\ell, u) \leq EW(\ell_0, u)$ pour toute loterie $\ell \in PO_U(L)$ (par définition de la $\succ_U$-dominance). D'après l'équation (5.1) définissant $PO_U(L)$, nous en déduisons que l'inégalité $EW(\ell, u) \leq EW(\ell_0, u)$ est vraie pour toute loterie $\ell \in L$. Ceci implique $\ell_0 \in PO_U(L)$ par définition de $PO_U(L)$ (cf. équation (5.1)), ce qui contredit notre hypothèse de départ. □

Proposition 44. *Pour tous ensembles de loteries L, L' :*

$$PO_U(L) \subseteq L' \subseteq L \Rightarrow PO_U(L') \subseteq PO_U(L)$$

Démonstration. Soient deux ensembles de loteries L, L' quelconques. Supposons que les inclusions suivantes sont vérifiées : $PO_U(L) \subseteq L' \subseteq L$. Soit $\ell' \in PO_U(L')$. Nous voulons montrer que nous avons forcément $\ell' \in PO_U(L)$. Soulignons que la loterie ℓ' appartient bien à L car nous avons $PO_U(L') \subseteq L' \subseteq L$ par hypothèse. Ainsi, pour montrer que $\ell' \in PO_U(L)$, il suffit de démontrer que $\neg(L'' \succ_U \ell')$ pour tout ensemble de loteries $L'' \subseteq L$ (cf. proposition 39). Montrons ce résultat par contraction en supposant qu'il existe un ensemble de loteries $L_0 \subseteq L$ tel que $L_0 \succ_U \ell'$.

Puisque $\ell' \in PO_U(L')$ par définition, alors nous avons $\neg(L'' \succ_U \ell')$ pour tout ensemble de loteries $L'' \subseteq L'$ (cf. proposition 39). Comme nous avons aussi $PO_U(L) \subseteq L'$ par hypothèse, alors nous avons $\neg(L'' \succ_U \ell')$ pour tout ensemble de loteries $L'' \subseteq PO_U(L)$. Par conséquent, puisque $L_0 \subseteq L$ et $L_0 \succ_U \ell'$, alors il existe forcément une loterie $\ell_0 \in L_0$ telle que $\ell_0 \notin PO_U(L)$. En utilisant la proposition 43, nous en déduisons que $PO_U(L) \succ_U \ell_0$ est forcément vrai. Comme par ailleurs nous avons $L_0 \succ_U \ell'$ par hypothèse, alors nous obtenons $PO_U(L) \cup L_0 \backslash \{\ell_0\} \succ_U \ell'$ par transitivité de la $\succ_U$-dominance (cf. proposition 42). Soit $L_1 = PO_U(L) \cup L_0 \backslash \{\ell_0\}$. Remarquons que par définition nous avons $PO_U(L) \subseteq L_1$. S'il existe une

loterie $\ell_1 \in L_1$ telle que $\ell_1 \notin PO_U(L)$, alors avec un raisonnement similaire nous pouvons montrer que $L_2 \succ_U \ell'$, où $L_2 = PO_U(L) \cup L_1 \backslash \{\ell_1\}$. En itérant ce raisonnement, nous pouvons construire une séquence emboîtée $L_1 \supset \ldots \supset L_k = PO_U(L)$ telle que $L_j \succ_U \ell'$ pour tout $j \in \{1, \ldots, k\}$; en particulier, nous avons $PO_U(L) \succ_U \ell'$. Comme $PO_U(L) \subseteq L'$ par hypothèse, alors nous en déduisons que $\ell' \notin PO_U(L')$. Cependant, ce dernier fait contredit nos hypothèses car nous $\ell' \in PO_U(L')$ par définition. □

Proposition 45. *L'algorithme 9 retourne exactement l'ensemble $PO_U(\mathcal{L}_\mathcal{A})$.*

Démonstration. Soit L_S l'ensemble de loteries en sortie de l'algorithme. Nous allons montrer que nous avons à la fois $PO_U(\mathcal{L}_\mathcal{A}) \subseteq L_S$ et $L_S \subseteq PO_U(\mathcal{L}_\mathcal{A})$.

- $PO_U(\mathcal{L}_\mathcal{A}) \subseteq L_S$ *:* en utilisant la proposition 41, nous déduisons que seules des sous-stratégies conduisant à des stratégies $\succ_U$-dominées dans $\mathcal{L}_\mathcal{A}$ sont supprimées durant la résolution (en ligne 13). Puisque $PO_U(\mathcal{L}_\mathcal{A})$ est exactement l'ensemble des loteries non $\succ_U$-dominées dans $\mathcal{L}_\mathcal{A}$ (d'après la proposition 39), alors nous avons bien $PO_U(\mathcal{L}_\mathcal{A}) \subseteq L_S$.
- $L_S \subseteq PO_U(\mathcal{L}_\mathcal{A})$ *:* cette inclusion découle de la proposition 44 en prenant $L = \mathcal{L}_\mathcal{A}$ et $L' = L_S$. En effet, nous venons de montrer que l'inclusion $PO_U(\mathcal{L}_\mathcal{A}) \subseteq L_S$ est vraie et, par ailleurs, nous avons $L_S \subseteq \mathcal{L}_\mathcal{A}$. En appliquant la proposition 44, nous obtenons $PO_U(L_S) \subseteq PO_U(\mathcal{L}_\mathcal{A})$. Comme en plus nous avons $PO_U(L_S) = L_S$ (d'après la ligne 13, appliquée au nœud racine), alors nous en déduisons $L_S \subseteq PO_U(\mathcal{L}_\mathcal{A})$.

Ainsi, nous avons bien $L_S = PO_U(\mathcal{L}_\mathcal{A})$. □

Pour pouvoir mettre en œuvre cet algorithme, il est nécessaire de pouvoir réaliser l'étape de sélection des loteries en chaque nœud de décision D_k de manière efficace (cf. ligne 13). Pour toute loterie $\ell \in L_k$, s'il existe $L \subseteq L_k$ tel que $L \succ_U \ell$, alors nous avons forcément $L_k \succ_U \ell$. Par conséquent, pour savoir si une loterie $\ell \in L_k$ doit être éliminée ou non, il suffit de tester si $L_k \succ_U \ell$ est vraie ou non ; ainsi, il n'est pas nécessaire d'énumérer tous les ensembles $L \subseteq L_k$ pour pouvoir prendre cette décision. Par ailleurs, nous avons l'équivalence suivante :

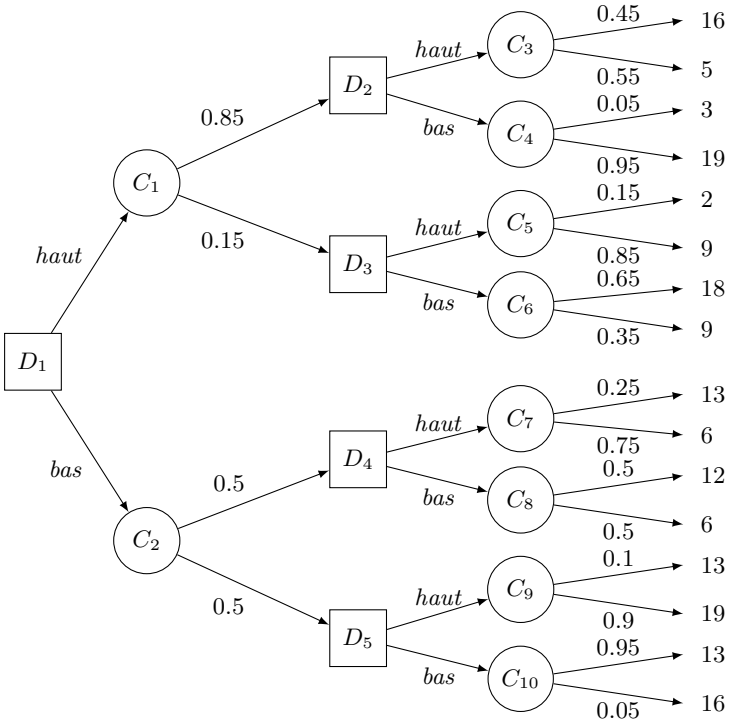
$$L_k \succ_U \ell \Leftrightarrow \min_{u \in U} \max_{\ell' \in L_k} \{EW(\ell', u) - EW(\ell, u)\} > 0$$

Ce résultat, qui découle directement de la définition 55, nous permet de tester si $L_k \succ_U \ell$ est vraie ou non en résolvant le problème d'optimisation suivant :

$$\begin{aligned}
\min \quad & t && (5.2)\\
\text{s.c.} \quad & t \geq EW(\ell', u) - EW(\ell, u),\ \forall \ell' \in L_k && (5.3)\\
& EU(\ell_1, u_i) - EU(\ell_2, u_i) \geq 0,\ \forall i \in N, \forall (\ell_1, \ell_2) \in \mathcal{P}_i && (5.4)\\
& u_i(x) \leq u_i(y),\ \forall i \in N, \forall x, y \in \mathbb{R} : x \leq y && (5.5)\\
& u_i(x) \in \mathbb{R},\ \forall i \in N, \forall x \in \mathbb{R} && (5.6)\\
& t \in \mathbb{R} && (5.7)
\end{aligned}$$

Dans ce problème d'optimisation, l'équation (5.3) est une linéarisation classique de l'opération de maximisation, l'équation (5.4) oblige la fonction d'utilité $u_i, i \in N$, à être compatible avec les données de préférences disponibles (stockées dans l'ensemble $\mathcal{P}_i$) et l'équation (5.5) garantie que la fonction u_i est croissante pour chaque agent $i \in N$. Notons que les variables de ce problème sont les fonctions d'utilité $u_i, i \in N$ (plus la variable t représentant le résultat de la maximisation). Dans la sous-section 5.4, nous proposerons une représentation des fonctions d'utilité $u_i, i \in N$, utilisant un ensemble fini de variables réelles, ce qui nous permettra de résoudre ce problème d'optimisation par programmation linéaire. L'exemple suivant présente une exécution de notre algorithme sur un problème simple :

Exemple 34. *Considérons une situation de décision séquentielle admettant la représentation suivante :*



Pour simplifier l'exemple, nous considérons que ce problème de décision séquentielle implique un seul agent que nous appelons décideur. *Supposons que le système ne possède aucune information sur les préférences du décideur. Dans ce cas, l'ensemble U est composé de toutes les fonctions $u : \mathbb{R} \to \mathbb{R}$ croissantes. Cet arbre de décision contient exactement 8 stratégies possibles conduisant à 8 loteries distinctes. Il s'agit pour nous de déterminer celles qui sont potentiellement optimales. Pour ce faire, nous déroulons maintenant l'algorithme 9.*

Durant la première itération, c'est le nœud D_5 qui est traité par l'algorithme. L'ensemble L_5 est constitué des loteries $(13, 0.1; 19, 0.9)$ et $(13, 0.95; 16, 0.05)$ associées respectivement aux sous-stratégies $\{(D_5, C_9)\}$ et $\{(D_5, C_{10})\}$. Comme $L_5 \succ_U (13, 0.95; 16, 0.05)$, alors la sous-stratégie $\{(D_5, C_{10})\}$ est éliminée au niveau du nœud D_5 (cf. ligne 13). Puisque nous avons $\neg(L_5 \succ_U (13, 0.1; 19, 0.9))$, alors nous avons $L_5.loteries = \{(13, 0.1; 19, 0.9)\}$ à la fin de cette itération (cf. ligne 13).

Durant la deuxième itération, c'est le nœud D_4 qui est considéré par l'algorithme. L'ensemble L_4 est composé des loteries $(6, 0.75; 13, 0.25)$ *et* $(6, 0.5; 12, 0.5)$ *associées respectivement aux sous-stratégies* $\{(D_4, C_7)\}$ *et* $\{(D_4, C_8)\}$*. Au niveau de ce nœud, la sous-stratégie* $\{(D_4, C_8)\}$ *est éliminée car nous avons* $L_4 \succ_U (6, 0.5; 12, 0.5)$ *(cf. ligne 13). En revanche, la sous-stratégie* $\{(D_4, C_7)\}$ *n'est pas supprimée car nous avons* $\neg(L_4 \succ_U (6, 0.75; 13, 0.25))$*. Par conséquent, nous avons* $L_4.loteries = \{(6, 0.75; 13, 0.25)\}$*.*

Durant la troisième itération, l'algorithme s'occupe du nœud D_3. L'ensemble L_3 est constitué des loteries $(2, 0.15; 9, 0.85)$ *et* $(9, 0.35; 18, 0.65)$ *associées respectivement aux sous-stratégies* $\{(D_3, C_5)\}$ *et* $\{(D_3, C_6)\}$*. Dans ce nœud, la sous-stratégie* $\{(D_3, C_5)\}$ *est éliminée car nous avons* $L_3 \succ_U (2, 0.15; 9, 0.85)$ *(cf. ligne 13). Par ailleurs, comme nous avons* $\neg(L_3 \succ_U (9, 0.35; 18, 0.65))$*, alors la sous-stratégie* $\{(D_3, C65)\}$ *n'est pas supprimée. Finalement, nous avons* $L_3.loteries = \{(9, 0.35; 18, 0.65)\}$*.*

Durant la quatrième itération, c'est le nœud D_2 qui est examiné par l'algorithme. L'ensemble L_2 est composé des loteries $(5, 0.55; 16, 0.45)$ *et* $(3, 0.05; 19, 0.95)$ *associées respectivement aux sous-stratégies* $\{(D_2, C_3)\}$ *et* $\{(D_2, C_4)\}$*. Comme nous avons* $\neg(L_2 \succ_U (5, 0.55; 16, 0.45))$ *et* $\neg(L_2 \succ_U (3, 0.05; 19, 0.95))$*, alors aucune sous-stratégie n'est éliminée en ce nœud et par conséquent nous avons* $D_2.loteries = \{(5, 0.55; 16, 0.45), (3, 0.05; 19, 0.95)\}$*.*

À la dernière itération, notre algorithme se concentre sur le nœud D_1 (la racine de l'arbre). L'ensemble L_1 est constitué des trois loteries suivantes :

- $\ell_1 = (5, 0.4675; 9, 0.0525; 16, 0.3825; 18, 0.0975)$ *associée à la stratégie* $\{(D_1, C_1), (D_2, C_3), (D_3, C_6)\}$*,*
- $\ell_2 = (3, 0.0425; 9, 0.0525; 18, 0.0975; 19, 0.8075)$ *associée à la stratégie* $\{(D_1, C_1), (D_2, C_4), (D_3, C_6)\}$*,*
- $\ell_3 = (6, 0.375; 13, 0.175; 19, 0.45)$ *associée à la stratégie* $\{(D_1, C_2), (D_4, C_7), (D_5, C_9)\}$*.*

Au niveau du nœud D_1, aucune stratégie n'est supprimée car nous avons $\neg(L_1 \succ_U \ell_1)$*,* $\neg(L_1 \succ_U \ell_2)$ *et* $\neg(L_1 \succ_U \ell_3)$ *(cf. ligne 13). Ainsi, nous avons* $L_1.loteries = \{\ell_1, \ell_2, \ell_3\}$*. Finalement, l'algorithme 9 se termine en retournant l'ensemble de loteries* $\{\ell_1, \ell_2, \ell_3\}$ *(cf. ligne 15) qui est composé de toutes les loteries potentiellement optimales du problème.*

L'algorithme 9 repose sur les bonnes propriétés de la relation $\succ_U$ qui permettent de construire efficacement les loteries potentiellement optimales dans un arbre de décision par programmation dynamique. Cependant, lorsque l'ensemble U des profils admissibles n'est pas suffisamment discriminant, l'ensemble $PO_U(\mathcal{L}_\mathcal{A})$ des loteries potentiellement optimales peut être très grand. En particulier, cet ensemble peut inclure $\Theta(2^{2^p})$ éléments pour des arbres de décision binaires parfaits de profondeur p (e.g., [Jeantet, 2010]). Pour de telles instances, l'algorithme 9 retourne un ensemble de loteries de taille exponentielle en le nombre de nœuds de l'arbre, ce qui ne permet pas de formuler une recommandation aux agents. Ceci nous motive à proposer dans la section suivante un algorithme pour la détermination d'une loterie nécessairement optimale par élicitation incrémentale des préférences.

5.3 Un algorithme interactif par induction arrière pour déterminer une solution optimale au sens de l'utilité sociale espérée

Dans la section précédente, nous avons proposé un algorithme pour la détermination de l'ensemble $PO_U(\mathcal{L}_\mathcal{A})$ des loteries potentiellement optimales dans l'arbre $\mathcal{A}$ (cf. algorithme 9). Lorsque les agents sont prêts à répondre à nos questions, nous pouvons envisager de les interroger sur leurs préférences individuelles afin de réduire suffisamment l'ensemble U de manière à détecter une loterie *nécessairement optimale* : la loterie $\ell \in \mathcal{L}_\mathcal{A}$ est dite nécessairement optimale si et seulement si $EW(\ell, u) \geq EW(\ell', u)$ pour toute loterie $\ell' \in \mathcal{L}_\mathcal{A}$ et tout profil admissible $u \in U$. Notons toutefois que l'ensemble L_S retourné par l'algorithme 9 contient forcément une loterie nécessairement optimale quand il existe une loterie $\ell \in L_S$ telle que $EW(\ell, u) \geq EW(\ell', u)$ pour toute loterie $\ell' \in L_S$ et tout profil admissible $u \in U$. En effet, par définition de $PO_U(\mathcal{L}_\mathcal{A})$, nous en déduisons que l'inégalité $EW(\ell, u) \geq EW(\ell', u)$ est vraie pour toute loterie $\ell' \in \mathcal{L}_\mathcal{A}$ et tout profil admissible $u \in U$, ce qui signifie que ℓ est nécessairement optimale. Ainsi, si l'ensemble L_S contient une telle loterie ℓ, alors il est inutile de poser des questions aux agents : cette loterie peut être recommandée aux agents sans craindre de regretter cette décision plus tard. Dans le cas contraire, nous pouvons envisager de poser des questions aux agents après l'exécution de l'algorithme L_S pour réduire progressivement l'ensemble U de manière à faire apparaître une loterie nécessairement optimale. Néanmoins, cette approche ne serait pas très efficace en pratique dans les situations où l'ensemble $PO_U(\mathcal{L}_\mathcal{A})$ contient un grand nombre d'éléments. C'est pourquoi nous proposons une nouvelle fois de mêler la recherche et l'élicitation incrémentale des préférences pour limiter à la fois les temps de résolution et le nombre de questions posées aux agents.

Afin de collecter des informations sur les préférences des agents, nous adoptons l'approche d'élicitation incrémentale fondée sur le critère Minimax Regret (cf. Section 1.4.2). Dans notre contexte, les notions de *pairwise max regret* (PMR), de *max regret* (MR) et de *minimax regret* (mMR) sont définies par :

$$\begin{aligned}
\mathrm{PMR}(\ell, \ell', U) &= \max_{u \in U} \Big\{ EW(\ell', u) - EW(\ell, u) \Big\} \\
\mathrm{MR}(\ell, \mathcal{L}_\mathcal{A}, U) &= \max_{\ell' \in \mathcal{L}_\mathcal{A}} \mathrm{PMR}(\ell, \ell', U) \\
\mathrm{mMR}(\mathcal{L}_\mathcal{A}, U) &= \min_{\ell \in \mathcal{L}_\mathcal{A}} \mathrm{MR}(\ell, \mathcal{L}_\mathcal{A}, U)
\end{aligned}$$

Par définition, la valeur $\mathrm{mMR}(\mathcal{L}_\mathcal{A}, U)$ représente la plus grande perte que nous pouvons induire en recommandant une loterie ℓ qui minimise la valeur $\mathrm{MR}(\ell, \mathcal{L}_\mathcal{A}, U)$. En particulier, si $\mathrm{mMR}(\mathcal{L}_\mathcal{A}, U) = 0$, alors toutes les loteries de l'ensemble $\arg\min_{\ell \in \mathcal{L}_\mathcal{A}} \mathrm{MR}(\ell, \mathcal{L}_\mathcal{A}, U)$ sont nécessairement optimales. Cette observation nous suggère de poser des questions aux agents tant que l'inégalité $\mathrm{mMR}(\mathcal{L}_\mathcal{A}, U) > \delta$ est vraie, où $\delta \geq 0$ est un seuil de tolérance que les agents estiment acceptable. Idéalement, nous aimerions nous renseigner auprès des agents tant que la valeur $\mathrm{mMR}(\mathcal{L}_\mathcal{A}, U)$ est strictement positive pour pouvoir identifier une loterie nécessairement optimale. Cependant, pour limiter le nombre de questions posées, il est préférable de considérer un seuil δ strictement positif représentant la perte maximale admissible.

Nous proposons maintenant un algorithme de recherche par élicitation incrémentale permettant de déterminer une loterie $\ell \in \mathcal{L}_\mathcal{A}$ telle que $\mathrm{MR}(\ell, \mathcal{L}_\mathcal{A}, U_f) \leq \delta$, où U_f représente l'ensemble des profils

d'utilités admissibles à la fin de l'exécution de l'algorithme. Notre algorithme par induction arrière pose des questions aux agents durant la résolution dans le but de discriminer entre les loteries stockées en chaque nœud de décision D_k (cf. algorithme 11). Plus précisément, nous engendrons des questions tant que $\text{mMR}(L_k, U) > \delta/d$, où d représente le plus grand nombre de nœuds de décision dans un chemin de la racine à une feuille de l'arbre $\mathcal{A}$. Contrairement à l'algorithme 9, une seule loterie est finalement stockée dans l'attribut *loteries* du nœud de décision D_k : cette loterie est choisie arbitrairement parmi les loteries ℓ qui minimisent la valeur $\text{MR}(\ell, L_k, U)$ (cf. lignes 20-21). Imposer $\text{mMR}(L_k, U) \leq \delta/d$ au lieu de $\text{mMR}(L_k, U) \leq \delta$ permet de garantir que le regret reste inférieur au seuil δ suite à la propagation des erreurs commises en chaque nœud de décision.

Algorithm 11: Induction arrière interactive

Input: $\mathcal{A} = (\mathcal{V}, \mathcal{E})$: un arbre de décision ; $U = U_1 \times \ldots \times U_n$: un ensemble de profils admissibles
Output: Une loterie nécessairement optimale

1 **for** $k = |\mathcal{V}_D|, \ldots, 1$ **do**
2 $L_k \leftarrow \emptyset$
3 **forall** $(D_k, A) \in \mathcal{E}$ **do**
4 **if** $A \in \mathcal{V}_D$ **then**
5 $L_k \leftarrow L_k \cup A.loteries$
6 **else if** $A \in \mathcal{V}_C$ **then**
7 $L_k \leftarrow L_k \cup \text{Combinaison}(A, \mathcal{A})$
8 **else**
 `/* cas où` A `est une feuille de l'arbre */`
9 $\ell \leftarrow (A, 1)$
10 $L_k \leftarrow L_k \cup \{\ell\}$
11 **end**
12 **end**
13 $L_k \leftarrow \{\ell \in L_k : \forall L \subseteq L_k, \neg(L \succ_U \ell)\}$
14 **while** $\text{mMR}(L_k, U) > \delta/d$ **do**
15 $(x, i) \leftarrow \text{Choisir}(\bigcup_{\ell \in L_k} S(\ell), N)$
16 $\lambda \leftarrow (\max_{u_i \in U_i} u_i(x) + \min_{u_i \in U_i} u_i(x))/2$
17 Demander à l'agent i de comparer les loteries ℓ^x et ℓ^λ
18 Mettre à jour l'ensemble U_i en fonction de sa réponse
19 **end**
20 Sélectionner une loterie ℓ dans l'ensemble $\arg\min_{\ell' \in L_k} \text{MR}(\ell', L_k, U)$;
21 $D_k.lotteries \leftarrow \{\ell\}$;
22 **end**
23 **return** $D_1.loteries$

Pour réduire la valeur $\text{mMR}(L_k, U)$, nous pourrions envisager de demander aux agents de comparer des loteries de l'ensemble L_k, comme dans la stratégie CSS (présentée en Section 1.4.2). Cependant, les loteries associées à des sous-stratégies dans un arbre de décision sont des objets complexes avec plusieurs conséquences, ce qui rend toute comparaison directe cognitivement difficile. Nous proposons à la place de demander aux agents de comparer des loteries certaines de type $\ell^x = (x, 1)$ avec des

loteries de type $\ell^\lambda = (x^\top, \lambda; x^\bot, 1-\lambda)$, où $x^\top$ et $x^\bot$ représentent respectivement la meilleure et la pire conséquence possible (cf. ligne 17). Soulignons le fait que demander de comparer des loteries certaines à des loteries de type ℓ^λ fait partie des questions standards dans la littérature sur l'aide à la décision [Keeney and Raiffa, 1976] ; ce type de questions a d'ailleurs été utilisé plus récemment dans de nombreux travaux (e.g., [Chajewska and Koller, 2000, Boutilier, 2002, Wang and Boutilier, 2003, Boutilier et al., 2006, Braziunas and Boutilier, 2007, 2010]).

Pour faciliter l'élicitation, nous pouvons imposer $u_i(x^\bot) = 0$ et $u_i(x^\top) = 1$ pour tout agent $i \in N$ puisque les utilités de vNM sont uniques à une transformation affine strictement croissante près. En procédant ainsi, nous obtenons $EU(\ell^\lambda, u_i) = \lambda u_i(x^\top) + (1-\lambda)u_i(x^\bot) = \lambda$ pour tout $\lambda \in [0,1]$. Par ailleurs, nous avons $EU(\ell^x, u_i) = u_i(x)$ pour toute conséquence $x \in \mathcal{V}_C$. Par conséquent, si l'agent i préfère la loterie ℓ^x à la loterie ℓ^λ, alors nous devons restreindre l'ensemble U_i aux fonctions u_i vérifiant la contrainte $u_i(x) \geq \lambda$; pour la préférence inverse, c'est la contrainte $u_i(x) \leq \lambda$ qui doit être imposée. Ceci montre que nous devons choisir $\lambda = (\max_{u_i \in U_i} u_i(x) + \min_{u_i \in U_i} u_i(x))/2$ pour réduire l'incertitude sur la valeur $u_i(x)$ le plus possible dans le pire scénario de réponse (cf. ligne 16). À chaque itération, la procédure Choisir sélectionne une conséquence x et un agent i pour lui demander de comparer les loteries ℓ^x et ℓ^λ (cf. ligne 15). Dans le but de réduire le plus possible la valeur $\mathrm{mMR}(L_k, U)$, cette procédure choisit une conséquence $x \in \bigcup_{\ell \in L_k} S(\ell)$ et un agent $i \in N$ tels que la valeur $\max_{u_i \in U_i} u_i(x) - \min_{u_i \in U_i} u_i(x)$ est la plus grande possible, où $S(\ell)$ est l'ensemble des conséquences possibles $\{x_1, \ldots, x_m\}$ de la loterie $\ell = (x_1, p_1; \ldots; x_m, p_m)$; l'ensemble $S(\ell)$ est souvent appelé le *support* de la loterie ℓ.

Pour pouvoir utiliser cette procédure, il est nécessaire de pouvoir calculer efficacement la valeur $\mathrm{mMR}(L_k, U)$ à chaque itération. Par définition, cette quantité peut être obtenue en calculant préalablement la valeur $\mathrm{PMR}(\ell, \ell', U)$ pour chaque paire de loteries $(\ell, \ell') \in L_k^2$. Il s'agit alors de proposer une méthode efficace pour le calcul de la quantité $\mathrm{PMR}(\ell, \ell', U)$ pour une paire de loteries $(\ell, \ell') \in L_k^2$ quelconque. Cette dernière quantité correspond à la valeur optimale du problème d'optimisation suivant :

$$\begin{aligned}
\max \quad & EW(\ell', u) - EW(\ell, u) && (5.8)\\
\text{s.c.} \quad & EU(\ell_1, u_i) - EU(\ell_2, u_i) \geq 0,\ \forall i \in N, \forall (\ell_1, \ell_2) \in \mathcal{P}_i && (5.9)\\
& u_i(x) \leq u_i(y),\ \forall i \in N, \forall x, y \in \mathbb{R} : x \leq y && (5.10)\\
& u_i(x) \in \mathbb{R},\ \forall i \in N, \forall x \in \mathbb{R} && (5.11)
\end{aligned}$$

Dans ce programme, il convient d'ajouter les conditions $u_i(x^\bot) = 0$ et $u_i(x^\top) = 1$ pour que les contraintes de l'équation (5.9) associées aux réponses des agents s'écrivent bien sous la forme "$u_i(x) \geq \lambda$" ou "$u_i(x) \leq \lambda$". Soulignons que, dans ce programme, les variables sont les fonctions d'utilité u_i, $i \in N$. Dans la section suivante, nous montrerons comment ces fonctions d'utilité peuvent être représentées de manière compacte pour que ce problème d'optimisation se formule comme un programme linéaire pouvant être résolu en temps polynomial. Mais avant cela, nous nous intéressons aux garanties de performance de notre procédure interactive, en supposant que ces calculs peuvent être effectués en temps polynomial.

Proposition 46. *Pour tout ensemble U de profils d'utilités admissibles et tout ensemble L de loteries :*

$$\Big[\forall i \in N, \forall x \in \bigcup_{\ell \in L} S(\ell),\ \max_{u_i \in U_i} u_i(x) - \min_{u_i \in U_i} u_i(x) \leq \delta/(2nd)\Big] \Rightarrow \mathrm{mMR}(L, U) \leq \delta/d$$

Démonstration. Notons u^* le profil d'utilité $(u_1^*, \ldots, u_n^*)$, où u_i^* représente la véritable fonction d'utilité de vNM de l'agent i. Soit $\ell = (x_1, p_1; \ldots; x_m; p_m)$ une loterie de l'ensemble $\arg\max_{\ell' \in L} EW(\ell', u^*)$. Notre objectif est de montrer que nous avons $\mathrm{MR}(\ell, L, U) \leq \delta/d$. Pour toute loterie $\ell' = (x'_1, p'_1; \ldots; x'_l; p'_l) \in L$:

$$\begin{aligned}
&\mathrm{PMR}(\ell, \ell', U) + EW(\ell, u^*) - EW(\ell', u^*) \\
&\quad = \max_{u \in U} \Big\{ \sum_{k=1}^{l} \Big(\sum_{i=1}^{n} u_i(x'_k) \Big) p'_k - \sum_{k=1}^{m} \Big(\sum_{i=1}^{n} (u_i(x_k) \Big) p_k \Big\} + \sum_{k=1}^{m} \Big(\sum_{i=1}^{n} u_i^*(x_k) \Big) p_k - \sum_{k=1}^{l} \Big(\sum_{i=1}^{n} u_i^*(x'_k) \Big) p'_k \\
&\quad = \max_{u \in U} \Big\{ \sum_{k=1}^{l} \Big(\sum_{i=1}^{n} (u_i(x'_k) - u_i^*(x'_k)) \Big) p'_k - \sum_{k=1}^{m} \Big(\sum_{i=1}^{n} (u_i(x_k) - u_i^*(x_k)) \Big) p_k \Big\} \\
&\quad \leq \sum_{k=1}^{l} \Big(n \times \frac{\delta}{2nd} \times p'_k \Big) - \sum_{k=1}^{m} \Big(n \times (-\frac{\delta}{2nd}) \times p_k \Big) \\
&\quad = \frac{\delta}{2d} \Big(\sum_{k=1}^{l} p'_k + \sum_{k=1}^{m} p_k \Big) \\
&\quad = \frac{\delta}{2d} \times 2 \\
&\quad = \frac{\delta}{d}
\end{aligned}$$

Ainsi, nous avons $\mathrm{PMR}(\ell, \ell', U) + EW(\ell, u^*) - EW(\ell', u^*) \leq \delta/d$. Comme $EW(\ell, u^*) - EW(\ell', u^*) \geq 0$ par définition de la loterie ℓ, alors nous en déduisons $\mathrm{PMR}(\ell, \ell', U) \leq \delta/d$ et donc $\mathrm{MR}(\ell, L, U) \leq \delta/d$. □

Proposition 47. *L'algorithme 11 a une complexité temporelle en $O(poly(n, |\mathcal{V}|, 1/\delta))$ et pose seulement $O(poly(n, |\mathcal{V}|, 1/\delta))$ questions dans le pire cas, où n est le nombre d'agents, $|\mathcal{V}|$ est le nombre de nœuds de l'arbre de décision et $\delta > 0$ est le seuil de tolérance.*

Démonstration. Concernant le nombre de questions : Initialement, puisque $u_i \in U_i$ est une fonction croissante pour tout $i \in N$, alors nous avons $\max_{u_i \in U_i} u_i(x) - \min_{u_i \in U_i} u_i(x) \leq u_i(x^\top) - u_i(x^\perp) = 1$ pour toute conséquence $x \in \mathcal{V}_C$. Notre procédure pose des questions au niveau de chaque nœud D_k tel que $\mathrm{mMR}(L_k, U) > \delta/d$. À chaque itération, la procédure Choisir sélectionne une conséquence $x \in \bigcup_{\ell \in L_k} S(\ell)$ et un agent $i \in N$ tels que la valeur $\max_{u_i \in U_i} u_i(x) - \min_{u_i \in U_i} u_i(x)$ est la plus grande possible. Comme nous avons $\mathrm{mMR}(L_k, U) > \delta/d$, alors nous avons forcément $\max_{u_i \in U_i} u_i(x) - \min_{u_i \in U_i} u_i(x) > \delta/2nd$ (d'après la proposition 46). La question posée en ligne 17 permet alors de diviser par deux la valeur $\max_{u_i \in U_i} u_i(x) - \min_{u_i \in U_i} u_i(x)$. Par conséquent, pour chaque conséquence $x \in \mathcal{V}_C$, notre algorithme a besoin de poser au plus $\lceil \log_2(2nd/\delta) \rceil$ questions par agent pour que la quantité $\max_{u_i \in U_i} u_i(x) - \min_{u_i \in U_i} u_i(x)$ passe de 1 à une valeur inférieure à $\delta/2nd$ pour tout $i \in N$. D'après la proposition 46, cela suffit à garantir que $\mathrm{mMR}(L_k, U) \leq \delta/d$ en tout nœud de décision D_k. Comme nous avons $|\mathcal{V}_C| \leq |\mathcal{V}|$ et $d \leq |\mathcal{V}_D| \leq |\mathcal{V}|$, alors nous en déduisons que notre algorithme pose au plus $n \times |\mathcal{V}| \times \lceil \log_2(2n|\mathcal{V}|/\delta) \rceil$.

Concernant la complexité temporelle : nous supposons ici que les tests de $\succ_U$-dominance et les calculs de PMR peuvent être réalisés en temps polynomial (par rapport à $|\mathcal{V}|$) par programmation linéaire (nous justifierons cette hypothèse dans la section suivante). À chaque nœud de décision D_k, l'algorithme 11 calcule l'ensemble L_k composé des loteries rattachées aux m_k enfants du nœud D_k (cf. lignes 2-12) ; les loteries $\succ_U$-dominées sont ensuite retirées de l'ensemble L_k (cf. ligne 13). Puisque une seule loterie est stockée dans l'attribut *loteries* de chaque nœud de décision, alors l'ensemble L_k contient au plus m_k éléments. Par conséquent, dans le pire cas, le calcul de $\text{mMR}(L_k, U)$ nécessite de résoudre $m_k(m_k - 1)$ problèmes PMR. Comme nous avons $m_k \leq |\mathcal{V}|$ et que les résolutions de PMR sont effectuées en temps polynomial (par rapport à $|\mathcal{V}|$), alors toutes les valeurs $\text{mMR}(L_k, U)$ sont obtenues en temps polynomial (par rapport à $|\mathcal{V}|$). Par ailleurs, à chaque nœud de décision D_k, la valeur $\text{mMR}(L_k, U)$ est calculée exactement $q_k + 1$ fois, où q_k est le nombre de questions posées à l'itération k. Comme nous venons de prouver que notre algorithme pose uniquement $O(poly(n, |\mathcal{V}|, 1/\delta))$ questions en tout et pour tout, alors celui-ci a une complexité temporelle en $O(poly(n, |\mathcal{V}|, 1/\delta))$. □

Ci-dessous, nous présentons le déroulement de l'algorithme 11 sur le problème de l'exemple 34 :

Exemple 35. *Déroulons l'algorithme 11 sur l'exemple 34 avec* $\delta = 0.001$. *Supposons que la fonction d'utilité de vNM du décideur est définie par* $u^*(x) = (x/20)^2$. *Supposons par ailleurs que le système ne possède aucune information sur les préférences du décideur avant de lancer la procédure de recherche interactive ; l'ensemble* U *est donc initialement composé de toutes les fonctions* $u : \mathbb{R} \to \mathbb{R}$ *croissantes. Dans l'exemple 34, nous avons constaté que les nœuds* D_5, D_4 *et* D_3 *contiennent une seule loterie non* $\succ_U$*-dominée. Par conséquent, l'algorithme 11 ne pose aucune question au niveau de ces nœuds. Les premières différences avec l'algorithme 9 apparaissent à la quatrième itération.*

Durant la quatrième itération, c'est le nœud D_2 *qui est examiné. L'ensemble* L_2 *est constitué des loteries* $(5, 0.55; 16, 0.45)$ *et* $(3, 0.05; 19, 0.95)$ *associées respectivement aux sous-stratégies* $\{(D_2, C_3)\}$ *et* $\{(D_2, C_4)\}$. *Comme* $\neg(L_2 \succ_U (5, 0.55; 16, 0.45))$ *et* $\neg(L_2 \succ_U (3, 0.05; 19, 0.95))$, *alors aucune sous-stratégie n'est éliminée ici (cf. ligne 13). En revanche, comme* $\text{mMR}(L_2, U) > \delta/2$, *alors le décideur doit répondre à une question (cf. ligne 14). La procédure Choisir retourne une conséquence* $x \in \{3, 5, 16, 19\}$ *qui maximise la valeur* $\max_{u \in U} u(x) - \min_{u \in U} u(x)$. *Comme* U *contient toutes les fonctions* u_i *croissantes, alors Choisir sélectionne arbitrairement dans l'ensemble* $\{3, 5, 16, 19\}$. *Supposons que Choisir retourne la conséquence* $x = 5$. *Dans ce cas, le décideur doit comparer la loterie* $(5, 1)$ *à la loterie* $(x^\top, 0.5; x^\perp, 0.5)$, *où* $x^\perp = 0$ *et* $x^\top = 20$ *(cf. ligne 17). Comme* $EU((5, 1), u^*) = 0.0625 < EU((x^\top, 0.5; x^\perp, 0.5), u^*) = 0.5$, *alors le décideur nous apprend qu'il préfère la loterie* $(x^\top, 0.5; x^\perp, 0.5)$ *à la loterie* $(5, 1)$. *L'ensemble* U *est ensuite mis à jour selon la contrainte* $u(5) \leq 0.5$ *(cf. ligne 18). À présent, nous avons* $\text{MR}((3, 0.05; 19, 0.95), L_2, U) \leq \delta/2$. *Par conséquent, la loterie* $(5, 0.55; 16, 0.45)$ *est éliminée et nous avons* $D_2.loteries = \{(3, 0.05; 19, 0.95)\}$ *(cf. ligne 21). Durant la dernière itération, notre algorithme s'occupe du nœud* D_1 *(la racine de l'arbre). L'ensemble* L_1 *est constitué des deux loteries suivantes :*

- $\ell_2 = (3, 0.0425; 9, 0.0525; 18, 0.0975; 19, 0.8075)$ *associée à la stratégie* $\{(D_1, C_1), (D_2, C_4), (D_3, C_6)\}$,
- $\ell_3 = (6, 0.375; 13, 0.175; 19, 0.45)$ *associée à la stratégie* $\{(D_1, C_2), (D_4, C_7), (D_5, C_9)\}$.

Comme $L_1 \succ_U \ell_3$*, alors la stratégie* $\{(D_1, C_1), (D_2, C_4), (D_3, C_6)\}$ *est retirée de* L_1 *(cf. ligne 13). Par conséquent, nous avons trivialement* $\text{mMR}(L_1, U) = 0 \leq \delta/2$ *et donc aucune question n'est posée ici (cf. ligne 14). Cette itération se termine en stockant la loterie* ℓ_3 *dans l'attribut* loteries *du nœud* D_1*. Finalement, notre algorithme retourne la loterie* ℓ_3 *qui est associée à la stratégie* $\{(D_1, C_2), (D_4, C_7), (D_5, C_9)\}$ *(cf. ligne 23). Le problème a ici été résolu en posant seulement une question au décideur.*

5.4 Représentation des utilités de von Neuman et Morgenstern

Pour pouvoir mettre en œuvre notre algorithme interactif introduit dans la section précédente, il convient de représenter efficacement les fonctions d'utilité de vNM de nos agents. En effet, pour réaliser les tests de $\succ_U$-dominance et calculer les valeurs $\text{mMR}(L_k, U)$ en chaque nœud de décision, nous devons résoudre des problèmes d'optimisation (cf. problème (5.2-5.7) et problème (5.8-5.11)) dont les variables sont les fonctions d'utilité de vNM.

5.4.1 Une variable par conséquence possible

Dans l'optique de résoudre ces problèmes par programmation linéaire, nous pouvons envisager un instant de définir une variable $u_{i,r}$ par paire $(i, x_r) \in N \times \mathcal{V}_C$ représentant l'utilité de l'agent i pour la conséquence x_r. Avec cette représentation numérique, toute donnée $(\ell, \ell') \in \mathcal{P}_i$ modélisant la préférence "je préfère la loterie ℓ à la loterie ℓ'" se traduit par l'inégalité suivante :

$$\sum_{r=1}^{|\mathcal{V}_C|} p(x_r) u_{i,r} \geq \sum_{r=1}^{|\mathcal{V}_C|} p'(x_r) u_{i,r}$$

où $p(x_r)$ (resp. $p'(x_r)$) représente la probabilité de la conséquence x_r avec la loterie ℓ (resp. ℓ'). De ce fait, l'ensemble U_i des fonctions d'utilité admissibles pour l'agent i peut être représenté par un polyèdre convexe, caractérisé par les contraintes linéaires imposées par les données de l'ensemble $\mathcal{P}_i$. Cette représentation permet ainsi de tester la $\succ_U$-dominance et de calculer les valeurs $\text{mMR}(L_k, U)$ en utilisant la programmation linéaire. Signalons que cette représentation des utilités a déjà été utilisée dans de précédents travaux sur l'élicitation incrémentale fondée sur le critère Minimax Regret (e.g., Wang and Boutilier [2003]).

Cependant, cette représentation possède deux inconvénients majeurs. D'une part, le nombre de variables $u_{i,r}$ croît avec le nombre d'agents impliqués et avec le nombre de feuilles de l'arbre de décision. D'autre part, comme les utilités de vNM sont des fonctions croissantes par définition, alors il est nécessaire de contraindre ces variables à vérifier aussi :

$$u_{i,r} \leq u_{i,r+1}, \forall i \in N, \forall r \in \{1, \ldots, |\mathcal{V}_C| - 1\}$$

en supposant que les conséquences sont numérotées de sorte que nous ayons $x_1 \leq \ldots \leq x_{|\mathcal{V}_C|}$. Par conséquent, cette représentation numérique des fonctions d'utilité de vNM semble trop coûteuse pour

résoudre efficacement des problèmes de décision collective avec un nombre important d'agents et/ou un arbre de décision de grande taille. Pour pouvoir appliquer nos algorithmes sur de grandes instances, nous présentons ci-après une représentation plus compacte fondée sur les splines.

5.4.2 Représentation par combinaison de I-splines

Les splines sont des fonctions polynomiales par morceaux dont les éléments sont reliés les uns aux autres de manière particulièrement lisse (égalité des dérivées aux points de raccordement). Ces fonctions sont très largement utilisées en interpolation de données et en approximation de formes du fait de leur capacité à approximer des formes complexes (par ajustement de courbe, conception interactive de courbe...) tout en garantissant que des courbes lissent seront produites à partir de données lisses (e.g., [Beatty and Barsky, 1995]). L'utilisation de fonctions polynomiales par morceaux en régression non linéaire étend l'avantage des fonctions polynomiales en offrant une plus grande flexibilité, des effets locaux par modification de paramètres et la possibilité d'imposer des contraintes sur les fonctions estimées (e.g., [Ramsay, 1988]).

Une autre caractéristique importante des splines est que celles-ci peuvent être engendrées par combinaison linéaire de splines de base. Cette spécificité est particulièrement intéressante dans notre contexte car elle permet de réduire l'élicitation de splines (représentant les utilités de vNM) à la détermination de leurs poids dans la base de splines considérée. Une base de splines particulièrement attractive pour la régression non linéaire est la famille des M-splines [Ramsay, 1988]. Les M-splines d'ordre k sont des fonctions polynomiales par morceaux dont chaque morceau (non nul) est de degré $k-1$. Sur un intervalle $[a, b]$ quelconque, nous pouvons définir une base de M-splines d'ordre k avec m paramètres libres à l'aide d'une séquence de nœuds $t = (t_1, \ldots, t_{m+k})$ vérifiant les propriétés suivantes :

- $t_1 = \ldots = t_k = a$,
- $t_{m+1} = \ldots = t_{m+k} = b$,
- $t_l \leq t_{l+1}$ pour tout $l \in \{1, \ldots, m+k-1\}$,
- $t_l < t_{l+k}$ pour tout $l \in \{1, \ldots, m\}$.

Plus précisément, la base de M-splines construite à partir d'une séquence de nœuds $t = (t_1, \ldots, t_{m+k})$ contient exactement m éléments d'ordre k, notés $M_l(x; k, t), l \in \{1, \ldots, m\}$, et définis pour tout $x \in [a, b]$ par la formule de récurrence suivante :

- Si $k = 1$, alors :

$$M_l(x; k, t) = \begin{cases} \dfrac{1}{t_{l+k} - t_l} & \text{si } x \in [t_l, t_{l+k}[\\ 0 & \text{sinon} \end{cases}$$

- Sinon :

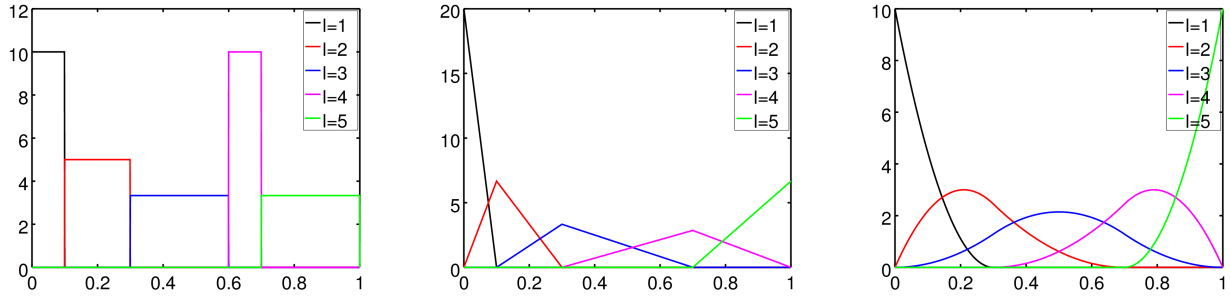
$$M_l(x; k, t) = \begin{cases} \dfrac{k[(x - t_l)M_l(x; k-1, t) + (t_{l+k} - x)M_{l+1}(x; k-1, t)]}{(k-1)(t_{l+k} - t_l)} & \text{si } x \in [t_l, t_{l+k}[\\ 0 & \text{sinon} \end{cases}$$

La séquence de nœuds t permet de définir la subdivision de l'intervalle $[a, b]$ telle que, sur chaque sous-intervalle, la spline $M_l, l \in \{1, \ldots, m\}$, est soit un polynôme de degré $k-1$ soit la fonction nulle. Par ailleurs, en tout point de raccordement x, le nombre de nœuds t_j dans la séquence t tels que $t_j = x$ définit l'ordre de continuité de chaque spline M_l au point x. En particulier, si un seul nœud t_j vérifie $t_j = x$, alors l'ordre de continuité en x de chaque spline M_l est égal à $k-1$; autrement dit les $k-2$ premières dérivées des morceaux incidents au point x associent la même image à la valeur x. Signalons que ce choix de conception est le plus rencontré en pratique pour la production de courbes lisses. L'existence de plusieurs nœuds t_j vérifiant $t_j = x$ entraîne un ordre de continuité plus faible au point de raccordement x. Par exemple, lorsque k nœuds t_j vérifient cette propriété, toute spline engendrée à partir de la base $M_l, l \in \{1, \ldots, m\}$, est discontinue au point de raccordement x (cf. [Ramsay, 1988] pour plus de détails).

Chaque spline M_l possède les mêmes propriétés qu'une densité de probabilité sur l'intervalle $[t_l, t_{l+k}[$. En particulier, M_l correspond à la densité uniforme pour $k = 1$ (cf. figure 5.1a) alors que M_l est la densité triangulaire de mode t_{l+1} pour $k = 2$ (cf. figure 5.1b). Plus généralement, la variable aléatoire X de distribution $M_l(x; k, t)$ a les moments [Ramsay, 1988] :

$$E[X] = \frac{\sum_{j=0}^{k} t_{l+j}}{k+1} \quad \text{et} \quad V[X] = \sum_{j=l}^{l+k} \sum_{j'=j+1}^{l+k} \frac{(t_{j'} - t_j)^2}{(k+1)^2(k+2)}$$

En pratique, il est assez standard d'utiliser des splines d'ordre $k = 3$ pour produire des courbes régulières tout en gardant suffisamment de flexibilité. En effet, ce choix permet à la fois d'engendrer des fonctions polynomiales par morceaux de degré relativement faible et de garantir que les dérivées premières et secondes de morceaux adjacents coïncident (cf. figure 5.1c pour un exemple avec $k = 3$).

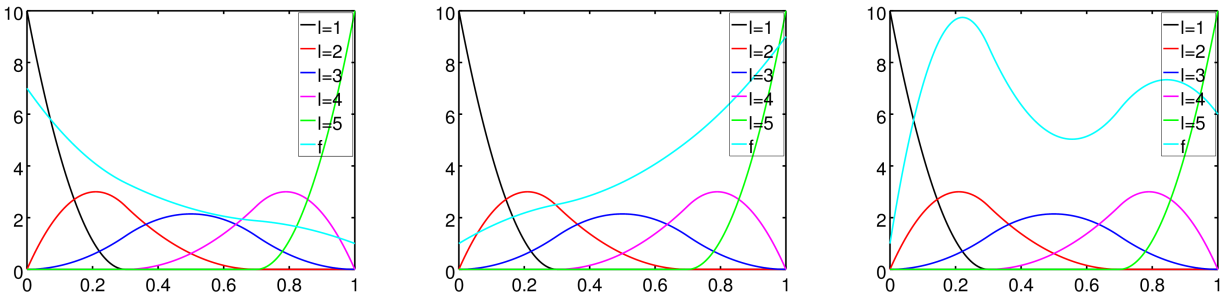


(a) $k=1$, $t=(0, 0.1, 0.3, 0.6, 0.7, 1)$. (b) $k=2$, $t=(0, 0, 0.1, 0.3, 0.7, 1, 1)$. (c) $k=3$, $t=(0, 0, 0, 0.3, 0.7, 1, 1, 1)$.

FIGURE 5.1 – Splines $M_l, l \in \{1, \ldots, m\}$, d'ordre k associées à la séquence t sur l'intervalle $[0, 1]$.

À partir d'une base de M-splines $M_l, l \in \{1, \ldots, m\}$, définie sur un intervalle $[a, b]$, nous pouvons approximer toute fonction définie sur $[a, b]$ par une spline de la forme $f = \sum_{l=1}^{m} \alpha_l M_l$ avec $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathbb{R}^m$. Comme nous avons $M_l(x; k, t) \geq 0$ pour tout $l \in \{1, \ldots, m\}$ et tout $x \in [a, b]$, alors la spline f est positive sur $[a, b]$ dès lors que les valeurs $\alpha_l, l \in \{1, \ldots, m\}$, sont toutes positives.

Par ailleurs, comme chaque M-spline $M_l, l \in \{1, \ldots, m\}$, est nulle en dehors de l'intervalle $[t_l, t_{l+k}[$, alors toute modification du poids α_l affecte uniquement la spline f sur ce dernier intervalle. Ceci illustre le fait que les M-splines offrent une certaine flexibilité locale très appréciable et une subdivision plus fine de l'intervalle $[a, b]$ permet d'obtenir un contrôle plus fin de la forme de la spline f. À titre illustratif, la figure 5.2 présente différentes splines f engendrées à partir d'une base de M-splines d'ordre 3.



(a) $\alpha = \{0.7, 1, 0.7, 0.4, 0.1\}$. (b) $\alpha = \{0.1, 0.5, 1, 1.5, 0.9\}$. (c) $\alpha = \{0.1, 3, 1, 2, 0.6\}$.

FIGURE 5.2 – Spline $f = \sum_{l=1}^{m} \alpha_l M_l$ (couleur cyan) pour la base de M-splines $M_l, l \in \{1, \ldots, m\}$, d'ordre 3 définie sur l'intervalle $[0, 1]$ avec $t = (0, 0, 0, 0.3, 0.7, 1, 1, 1)$.

Pour représenter l'utilité de vNM de l'agent $i \in N$, nous pouvons donc envisager dans un premier temps d'utiliser la formulation suivante :

$$u_i = \sum_{l=1}^{m} \alpha_l^i M_l$$

où $\alpha_l^i, l \in \{1, \ldots, m\}$, sont les coefficients permettant de modéliser au mieux les préférences de l'agent i avec la base $M_l, l \in \{1, \ldots, m\}$. Le principal avantage de cette représentation, par rapport à celle de la Section 5.4.1, est de fournir une définition plus compacte des utilités. En effet, seules $m \times n$ variables sont ici nécessaires pour caractériser complètement le profil d'utilité $u = (u_1, \ldots, u_n)$ contre $|\mathcal{V}_C| \times n$ variables pour l'autre représentation.

Avec la représentation fondée sur les M-splines, toute donnée $(\ell, \ell') \in \mathcal{P}_i$ représentant la préférence "je préfère la loterie ℓ à la loterie ℓ'" se traduit par la contrainte linéaire suivante :

$$\sum_{r=1}^{|\mathcal{V}_C|} p(x_r) \sum_{l=1}^{m} \alpha_l^i M_l(x_r; k, t) \geq \sum_{r=1}^{|\mathcal{V}_C|} p'(x_r) \sum_{l=1}^{m} \alpha_l^i M_l(x_r; k, t)$$

où $p(x_r)$ (resp. $p'(x_r)$) représente la probabilité de la conséquence x_r avec la loterie ℓ (resp. ℓ'). Cependant, ces contraintes ne suffisent pas à caractériser exactement l'ensemble U_i des fonctions u_i admissibles pour l'agent i car rien ne garantit que u_i est bien une fonction croissante (cf. figure 5.2 pour des contre-

exemples). Il faut donc par ailleurs imposer les contraintes suivantes :

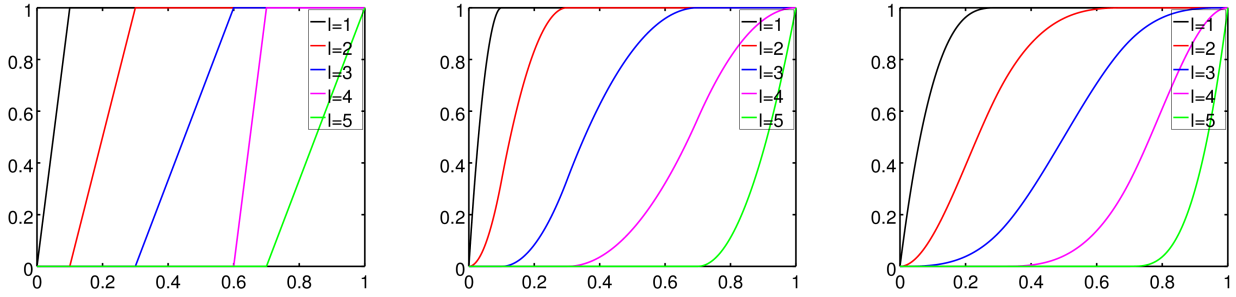
$$\sum_{l=1}^{m} \alpha_l^i M_l(x_r; k, t) \leq \sum_{l=1}^{m} \alpha_l^i M_l(x_{r+1}; k, t),\ \forall i \in N, \forall r \in \{1, \ldots, |\mathcal{X}_\mathcal{A}| - 1\}$$

en supposant que les conséquences sont numérotées de sorte que nous ayons $x_1 \leq \ldots \leq x_{|\mathcal{V}_C|}$. Par conséquent, cette représentation des préférences conduit à des formulations linéaires dont le nombre de contraintes augmente avec le nombre d'agents et le nombre de conséquences de l'arbre de décision.

Pour obtenir une représentation plus compacte des préférences, il nous faudrait en réalité une base de splines spécialement conçue pour la génération de fonctions non décroissantes. Remarquons que si toutes les splines de la base sont non décroissantes, alors toute combinaison linéaire de ces splines à coefficients positifs conduit nécessairement à une fonction non décroissante. Du fait que les M-splines sont positives par définition, il a été proposé de considérer la famille de fonctions obtenue par intégration des M-splines pour construire des bases de fonctions non décroissantes ; ces fonctions obtenues par intégration sont appelées I-splines (e.g., [Ramsay, 1988]). Plus formellement, à partir d'une base de M-splines $M_l, l \in \{1, \ldots, m\}$, nous obtenons la base de I-splines $I_l, l \in \{1, \ldots, m\}$, définies par :

$$I_l(x; k, t) = \int_a^x M_l(y; k, t)\, dy$$

Comme les M-splines sont des fonctions polynomiales par morceaux (non nuls) de degré $k-1$, alors les I-splines $I_l, l \in \{1, \ldots, m\}$, sont des fonctions polynomiales par morceaux (non nuls) de degré k ; la figure 5.3 nous montre les bases de I-splines associées aux bases de M-splines de la figure 5.1.



(a) $k=1$, $t=\{0, 0.1, 0.3, 0.6, 0.7, 1\}$. (b) $k=2$, $t=\{0, 0, 0.1, 0.3, 0.7, 1, 1\}$. (c) $k=3$, $t=\{0, 0, 0, 0.3, 0.7, 1, 1, 1\}$.

FIGURE 5.3 – Splines $I_l, l \in \{1, \ldots, m\}$, d'ordre k associées à la séquence t sur l'intervalle $[0, 1]$.

Chaque I-spline I_l admet par ailleurs, pour tout $x \in [a, b]$, la formulation alternative suivante :

$$I_l(x; k, t) = \begin{cases} 0 & \text{si } l > j \\ \displaystyle\sum_{l'=l}^{j} \frac{(t_{l'+k+1} - t_{l'}) M_{l'}(x; k+1, t)}{k+1} & \text{si } j-k+1 \leq l < j \\ 1 & \text{sinon} \end{cases}$$

où j est l'entier tel que $t_j \leq x < t_{j+1}$. Cette formulation nous donne un moyen pratique de calculer la valeur $I_l(x;k,t)$ à partir de la base de M-splines d'ordre $k+1$ définie avec la même séquence de nœuds t. Notons que, pour que cette formulation soit valide dans le cas où $j = m$, nous devons ajouter à la fin de la séquence de nœuds t un élément t_{m+k+1} égal à b.

À partir d'une base de I-splines $I_l, l \in \{1,\ldots,m\}$, nous pouvons approximer toute fonction non décroissante définie sur $[a,b]$ par une spline de la forme $f = \sum_{l=1}^{m} \alpha_l I_l$ avec $\alpha = (\alpha_1,\ldots,\alpha_m) \in \mathbb{R}_+^m$. En utilisant uniquement ces m valeurs $\alpha_l, l \in \{1,\ldots,m\}$, il est possible d'engendrer une infinité de fonctions lisses permettant d'approximer relativement bien toutes les fonctions non décroissantes sur $[a,b]$; pour les courbes complexes à irrégularités locales, il convient bien évidemment d'utiliser une subdivision fine de l'intervalle $[a,b]$. Récemment, une procédure d'élicitation incrémentale a été proposée pour le modèle RDU [Quiggin, 1992] (pour *Rank-Dependent Utility* en anglais) utilisant la famille des I-splines pour approximer la fonction de déformation des probabilités de ce modèle [Perny et al., 2016]. En ce qui nous concerne, il s'agit de représenter les fonctions d'utilité de vNM à l'aide de la famille de I-splines. Plus précisément, nous proposons de définir les utilités de vNM de nos agents par :

$$u_i = \sum_{l=1}^{m} \alpha_l^i I_l$$

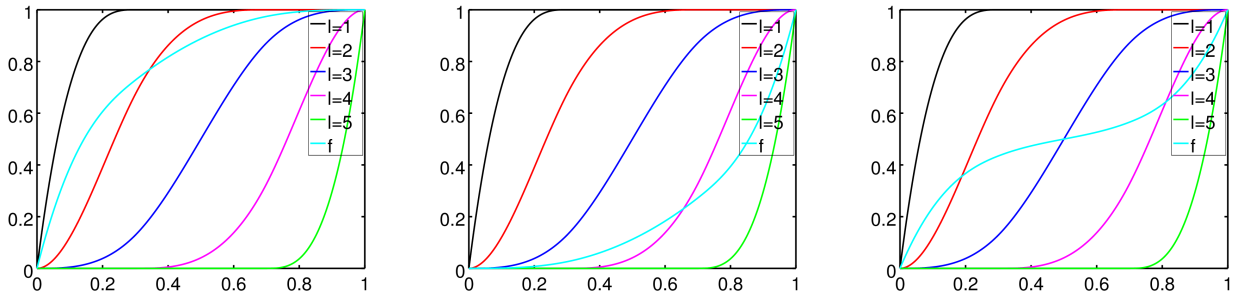
où $\alpha^i = (\alpha_1^i,\ldots,\alpha_m^i) \in \mathbb{R}_+^m$ est le vecteur de poids positifs permettant de représenter au mieux les préférences de l'agent $i \in N$. Avec cette représentation, le profil d'utilité $(u_1,\ldots,u_n)$ est complètement caractérisé avec seulement $m \times n$ paramètres. Par ailleurs, en utilisant cette représentation, l'ensemble U_i des utilités u_i admissibles est entièrement décrit par les contraintes suivantes :

$$\sum_{r=1}^{|\mathcal{V}_C|} p(x_r) \sum_{l=1}^{m} \alpha_l^i I_l(x_r;k,t) \geq \sum_{r=1}^{|\mathcal{V}_C|} p'(x_r) \sum_{l=1}^{m} \alpha_l^i I_l(x_r;k,t),\ \forall (\ell,\ell') \in \mathcal{P}_i$$

$$\alpha_l^i \geq 0,\ \forall l \in \{1,\ldots,m\}, \forall i \in N$$

où $p(x_r)$ (resp. $p'(x_r)$) représente la probabilité de la conséquence x_r avec la loterie ℓ (resp. ℓ'). Par conséquent, l'ensemble U_i forme un polyèdre convexe, ce qui permet d'utiliser la programmation linéaire pour réaliser les tests de $\succ_U$-dominance et calculer les valeurs $\text{mMR}(L_k,U)$. Notons que les I-splines nous offrent ici une représentation compacte des profils admissibles, utilisant un nombre de variables et de contraintes qui ne dépend pas de la taille de l'arbre.

La figure 5.4 présente trois fonctions f de différentes formes que nous avons engendrées à partir d'une base de I-splines d'ordre $k = 3$. Dans notre contexte décisionnel, ces fonctions pourraient très bien représenter les utilités de vNM u_i de trois agents impliqués dans le processus de décision. Dans ce cas, la figure 5.4a (resp. figure 5.4b) correspondrait à un agent adverse au risque (resp. attiré par le risque) car sa fonction d'utilité est concave (resp. convexe), tandis que la figure 5.4c serait associée à un agent dont les préférences sont définies par une utilité dite *en forme de S* (e.g., [Fishburn and Kochenberger, 1979, Kahneman and Tversky, 1979]).



(a) $\alpha = \{0.5, 0.3, 0.2, 0, 0\}$. (b) $\alpha = \{0, 0, 0.2, 0.3, 0.5\}$. (c) $\alpha = \{0.3, 0.2, 0, 0.2, 0.3\}$.

FIGURE 5.4 – Spline $f = \sum_{l=1}^{m} \alpha_l I_l$ (couleur cyan) pour la base de I-splines $I_l, l \in \{1, \ldots, m\}$, d'ordre 3 définie avec $t = \{0, 0, 0, 0.3, 0.7, 1, 1, 1\}$ sur l'intervalle $[0, 1]$.

Dans le cadre de cette thèse, nous travaillons uniquement avec des bases composées de I-splines d'ordre $k = 3$ et définies sur l'intervalle $[a, b] = [0, 1]$. Nous supposons donc que les conséquences de l'arbre de décision ont été normalisées au préalable de sorte que nous ayons $x^{\perp} = 0$ et $x^{\top} = 1$, où $x^{\perp}$ et $x^{\top}$ représentent respectivement la pire et la meilleure conséquence de l'arbre de décision. Rappelons que, dans la Section 5.3, nous avons décidé d'imposer les contraintes $u_i(x^{\perp}) = 0$ et $u_i(x^{\top}) = 1$ en utilisant le fait que les utilités de vNM sont uniques à une transformation affine strictement croissante près. Puisque les I-splines $I_l, l \in \{1, \ldots, m\}$, vérifient $I_l(0; k, t) = 0$ pour tout ordre k et toute séquence de nœuds t, alors nous avons $f(0) = 0$ pour toute spline f de la forme $f = \sum_{l=1}^{m} \alpha_l I_l$; par conséquent, nous avons forcément $u_i(0) = 0$ lorsque u_i est définie par $u_i = \sum_{l=1}^{m} \alpha_l^i I_l$. En revanche, pour avoir $u_i(1) = 1$, nous devons ajouter la contrainte de normalisation suivante :

$$\sum_{l=1}^{m} \alpha_l^i = 1$$

En effet, comme $I_l(1; k, t) = 1$ pour tout ordre k et toute séquence de nœuds t, alors cette normalisation implique $u_i(1) = \sum_{l=1}^{m} \alpha_l^i I_l(1; k, t) = \sum_{l=1}^{m} \alpha_l^i = 1$.

5.5 Résultats expérimentaux

Dans cette sous-section, nous présentons différents résultats expérimentaux permettant d'évaluer les performances de nos méthodes [2]. Pour commencer, nous comparons RI la représentation par I-spline (cf. Section 5.4.2) avec RP la représentation consistant à ajouter une variable par paire *(agent,conséquence)* (cf. Section 5.4.1) en terme de temps nécessaire au calcul de la valeur $\text{mMR}(\mathcal{L}, U)$, où U est l'ensemble de tous les profils possibles et $\mathcal{L}$ est un ensemble de loteries engendré aléatoirement. Pour obtenir des instances difficiles, l'ensemble $\mathcal{L}$ est engendré de sorte que toutes les loteries soient incomparables les

2. Les tests ont été réalisés sur un Intel Core i7 CPU 3.60GHz avec 16GB de mémoire et les optimisations linéaires ont été effectuées par le solveur Gurobi depuis un programme écrit en Java.

unes aux autres avec la dominance stochastique. Dans ces expériences, nous utilisons la base de I-splines d'ordre $k = 3$ représentée dans la figure 5.4. Par ailleurs, nous faisons varier n le nombre d'agents, m le nombre de loteries et c le nombre de conséquences à probabilité non nulle dans chaque loterie de l'ensemble $\mathcal{L}$. Les résultats obtenus sont donnés dans la table 5.1.

n	m	c	RP	RI
10	100	5	1.14	0.84
10	100	10	1.54	0.76
10	100	20	2.45	0.82
10	200	5	2.90	2.31
10	200	10	3.61	1.58
10	200	20	5.38	2.11
50	100	5	3.70	1.11
50	100	10	7.48	1.18
50	100	20	15.27	1.43
50	200	5	9.85	2.83
50	200	10	18.73	2.82
50	200	20	37.96	2.61

TABLE 5.1 – Temps nécessaire au calcul de la valeur mMR (moyenne sur 30 tests, en secondes).

Dans cette table, nous voyons que, dans toutes les configurations considérées, les temps de calcul sont meilleurs avec la représentation RI qu'avec la représentation RP. Par ailleurs, l'écart de performance est d'autant plus important que la taille de l'instance est grande. Par exemple, la représentation RI permet de réduire les temps de calcul par un facteur 1.4 sur les instances avec 10 agents, 100 loteries et 5 conséquences possibles, et par un facteur 15 sur les instances avec 50 agents, 200 loteries et 20 conséquences possibles. Par conséquent, la représentation RI semble plus adaptée que la représentation RP à la résolution de problèmes de décision collective sur domaine combinatoire.

Les expériences suivantes ont pour but d'évaluer la pertinence de notre stratégie de génération de questions, présentée en Section 5.3 et nommée IE ci-après, pour la détermination de la meilleure loterie dans un ensemble $\mathcal{L}$ de loteries donné en extension ; les ensembles $\mathcal{L}$ sont ici engendrés aléatoirement de manière à contenir 100 loteries à 5 conséquences possibles non dominées stochastiquement. Nous avons implémenté une version de IE avec la représentation RI (nommée IE-RI ci-après) et une version avec la représentation RP (nommée IE-RP ci-après). Comme base de comparaison, nous avons aussi implémenté la stratégie de génération de questions nommée IE-RP-Rand qui se distingue de IE-RP uniquement en choisissant aléatoirement l'agent i et la conséquence x à chaque itération. Pour chaque agent $i \in N$, nous commençons avec un ensemble $\mathcal{P}_i$ de données collectées vide et les réponses aux questions sont simulées à l'aide d'une fonction d'utilité u_i préalablement engendrée de manière aléatoire. Dans la figure 5.5, nous reportons les valeurs $\text{mMR}(\mathcal{L}, U)$ observées à chaque itération des procédures. Ces regrets sont exprimés sur une échelle normalisée, attribuant la valeur 1 à la valeur mMR initiale.

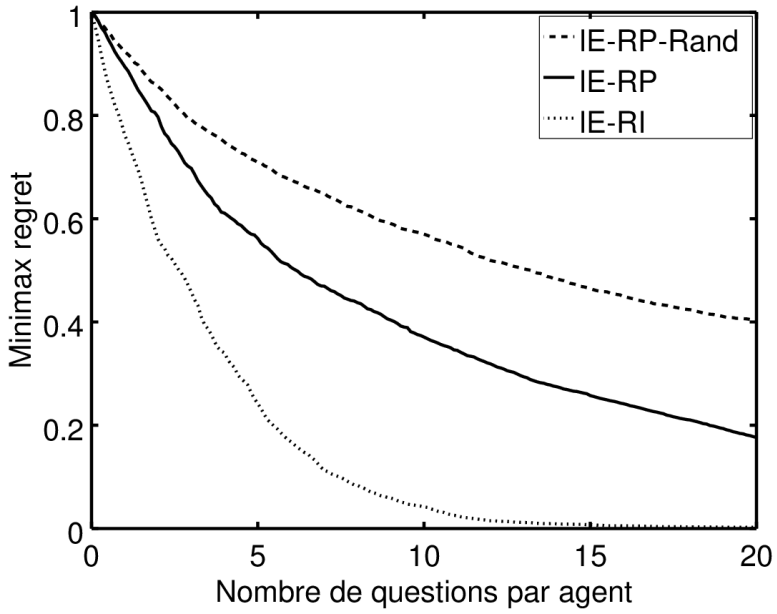


FIGURE 5.5 – Comparaison de méthodes d'élicitation incrémentale pour des problèmes de décision avec 10 agents et 100 loteries (30 tests).

En comparant IE-RP et IE-RP-Rand, nous voyons que la valeur $\text{mMR}(\mathcal{L}, U)$ réduit plus vite avec notre stratégie de génération de questions qu'avec la stratégie aléatoire fondée sur le même type de questions. En confrontant ensuite IE-RI et IE-RP, nous observons que la valeur $\text{mMR}(\mathcal{L}, U)$ diminue plus rapidement avec la représentation RI qu'avec la représentation RP. Par exemple, après 15 questions par agent en moyenne, IE-RI est capable de déterminer une loterie nécessairement optimale (car la valeur $\text{mMR}(\mathcal{L}, U)$ est à zéro) alors que la valeur $\text{mMR}(\mathcal{L}, U)$ est toujours au dessus de 30% du regret initial avec RP. Ceci s'explique par le fait que cette dernière représentation possède plus de paramètres à éliciter.

La figure 5.6 présente l'imprécision finale sur la fonction u_i de quatre agents à la fin de l'exécution de IE-RI sur une instance avec 100 loteries possibles. Sur cette figure, nous donnons la fonction u_i de chaque agent i (en trait continu) ainsi que les bornes supérieures et inférieures sur les valeurs $u_i(x)$ compatibles avec les données de préférence collectées : les lignes en pointillé (resp. tiretées) correspondent aux bornes calculées avec la représentation RI (resp. RP). Cette figure permet d'illustrer l'intérêt de l'approche incrémentale : même si les fonctions u_i ne sont pas connues avec précision, l'élicitation a été interrompue parce qu'une loterie nécessairement optimale a été détectée par le système. Par ailleurs, observons que les utilités des agents n'ont pas été élicitées avec le même degré de précision, ce qui ne nous a pas empêché de déterminer la meilleure loterie pour le groupe. Enfin, cette figure montre aussi la puissance des I-splines : l'imprécision sur les fonctions u_i est toujours plus faible avec la représentation RI qu'avec la représentation RP.

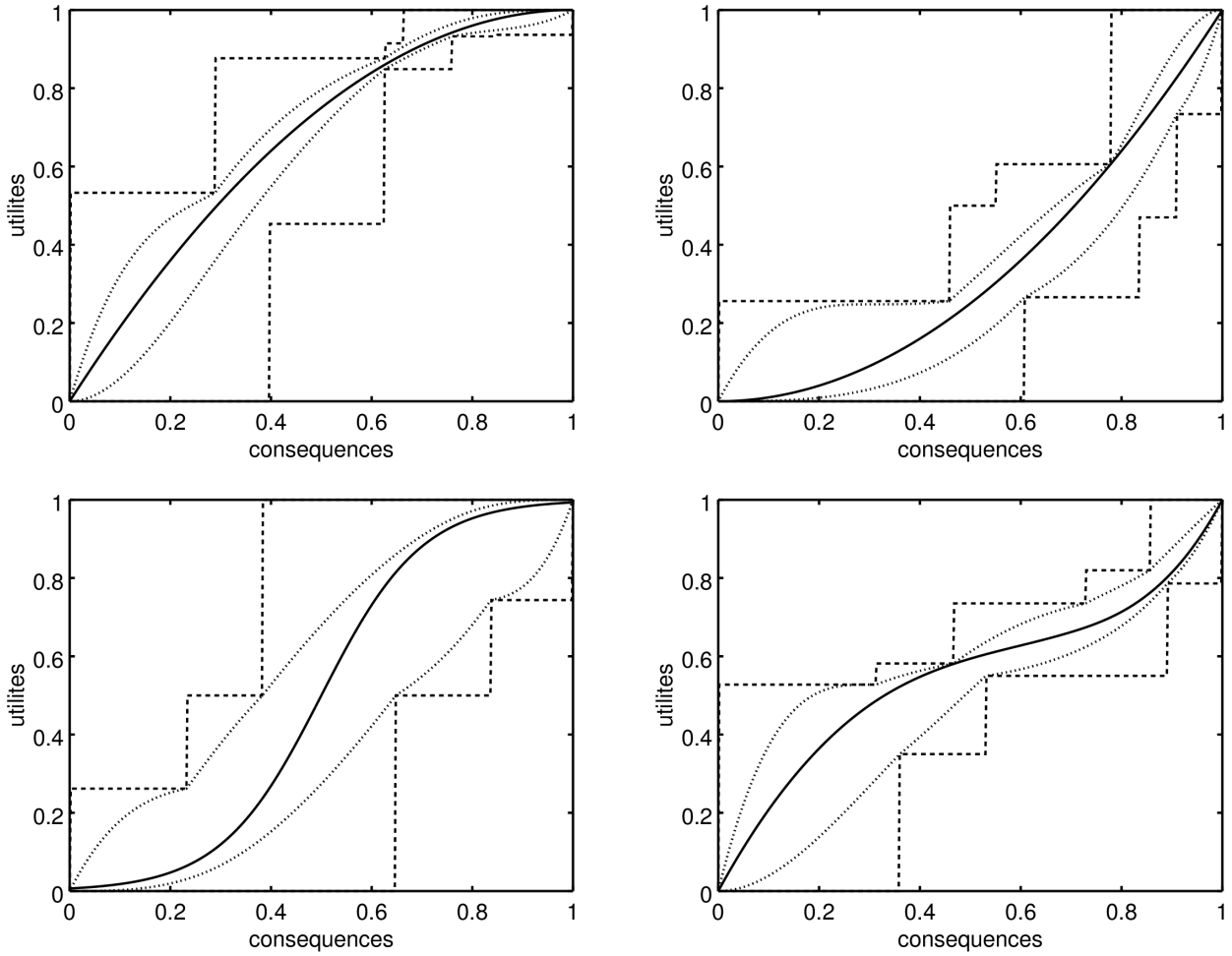


FIGURE 5.6 – Imprécision finale sur les valeurs d'utilité $u_i(x)$ observées à la fin de la procédure IE-RI sur une instance impliquant uniquement quatre agents.

Nos dernières expériences ont pour objectif de tester l'efficacité de notre algorithme par induction arrière interactive (nommée IAI ci-après) sur des ensembles de loteries définis implicitement par un arbre de décision. À des fins comparatives, nous considérons aussi la procédure en deux phases (nommée DP ci-après) consistant à calculer l'ensemble $PO_U(\mathcal{L}_\mathcal{A})$ avec l'algorithme 9 puis à appliquer la stratégie IE sur cet ensemble. Pour ces deux procédures, nous considérons la représentation RI qui utilise des I-splines. Comme dans les expériences précédentes, nous commençons avec un ensemble $\mathcal{P}_i$ vide pour tout agent i et nous simulons ses réponses avec une fonction u_i engendrée aléatoirement. Nous engendrons ici des arbres binaires parfaits, alternant nœuds de décision et nœuds de chance sur chaque chemin depuis la racine. Dans nos expériences, nous faisons varier la profondeur de l'arbre de décision de 6 à 20 pour évaluer l'impact de la taille de l'arbre. Dans la table 5.2, nous reportons à la fois le temps de résolution moyen (en secondes) et le nombre de questions moyen par agent. Dans cette table, le symbole "-" signifie tout simplement que la procédure continue de s'exécuter après 5 heures.

	IAI		DP	
profondeur	temps	questions	temps	questions
6	4.7	8.28	15.0	6.4
8	7.5	12.9	41.2	6.9
10	10.1	15.9	362.0	8.4
12	14.0	17.8	17475.1	9.7
14	27.1	20.0	-	-
16	73.1	21.3	-	-
18	264.1	22.4	-	-
20	1031.3	23.4	-	-

TABLE 5.2 – Comparaison des méthodes IAI et DP pour les arbres de décision ($n = 10$, $\delta = 0.1$, 30 tests)

Tout d'abord, nous voyons que la procédure IAI est très efficace sur les deux critères considérés (c'est-à-dire le temps d'exécution et le nombre de questions). Par exemple, sur nos plus grandes instances (profondeur égale à 20), IAI est capable de formuler une recommandation en moins de 25 minutes et avec moins de 25 questions par agent en moyenne. Par ailleurs, notons que IAI est beaucoup plus rapide que DP en pratique : avec une profondeur de 12, IAI se termine en moins de 15 secondes alors que DP a besoin de plus de 4 heures pour formuler une recommandation. Cette observation illustre le fait que combiner la recherche avec l'élicitation incrémentale des préférences permet d'accélérer les temps de résolution sur domaine combinatoire.

5.6 Conclusion du chapitre

Dans ce chapitre, nous avons étudié le potentiel de l'élicitation incrémentale dans le cadre de problèmes de décision collective dans le risque sur domaine combinatoire avec préférences incomplètes. Plus particulièrement, nous nous sommes intéressés aux problèmes de décision séquentielle se modélisant sous la forme d'arbre de décision et nous avons adopté le critère de l'utilité sociale espérée pour la comparaison des stratégies/loteries possibles. Dans un premier temps, nous avons proposé un algorithme de recherche par induction arrière pour la détermination de toutes les loteries potentiellement optimales au sens de l'utilité sociale espérée. Dans un second temps, nous avons introduit un algorithme combinant l'élicitation incrémentale et la recherche pour l'identification d'une loterie (presque) optimale. Nous avons montré que cet algorithme interactif de complexité polynomiale pose un nombre polynomial de questions dans le pire cas. Par ailleurs, cet algorithme s'est révélé très efficace en pratique comparé à une méthode en deux phases, et plus particulièrement lorsque les utilités de vNM sont représentées à l'aide de I-splines.

Conclusion et perspectives

Synthèse

Les travaux menés durant cette thèse relèvent de la théorie de la décision algorithmique, un domaine qui se situe au carrefour de la théorie de la décision, de l'intelligence artificielle et de l'optimisation combinatoire. Ces travaux portent essentiellement sur la prise de décision par élicitation incrémentale de préférences dans des contextes où les alternatives possibles sont évaluées selon différents points de vue (critères, agents, scénarios). Cette thèse débute ainsi par une introduction à la problématique de modélisation des préférences en décision multi-critère, multi-agents et dans le risque. Après une courte discussion portant sur les approches standards en élicitation des préférences, nous nous concentrons sur l'approche incrémentale fondée sur la notion de regret pour parvenir à déterminer la meilleure alternative du problème tout en limitant le nombre de questions posées aux agents.

Modélisation complexe des préférences. La première difficulté que nous nous sommes efforcés de surmonter réside dans l'utilisation de modèles de préférences sophistiqués, permettant de refléter des comportements décisionnels relativement complexes. Dans le cadre de problèmes de décision multicritère, nous nous sommes intéressés à l'intégrale de Choquet qui est une fonction d'agrégation très appréciée dans le domaine de l'aide à la décision, notamment pour sa capacité à modéliser des interactions positives et négatives entre plusieurs critères. L'utilisation d'une intégrale de Choquet dans une approche incrémentale pose un certain nombre de difficultés techniques liées aux paramètres de ce modèle décisionnel. En particulier, cela nécessite de pouvoir résoudre efficacement des problèmes d'optimisation comprenant un nombre de variables et de contraintes qui est exponentiel en le nombre de critères du problème. Dans le Chapitre 2, nous avons montré que ces problèmes d'optimisation pouvaient être simplifiés et résolus en temps polynomial, à condition de restreindre les interactions avec le décideur à un certain type de questions/réponses. Nous avons exploité ce résultat pour répondre aux trois problématiques de référence en aide à la décision : le choix (détermination de la meilleure alternative), le rangement (classement des alternatives de la meilleure à la plus mauvaise) et le tri (affectation des alternatives à des catégories prédéfinies). Dans le cadre de problèmes de décision multi-agents, nous nous sommes intéressés à la détermination de la meilleure alternative au sens de la méthode de Borda (cf. Chapitre 4). Cette dernière règle de vote, très connue dans le domaine du choix social, permet d'éviter des situations de "tyrannie de la majorité" : l'alternative gagnante par la méthode de Borda est généralement

moins controversée que le vainqueur de Condorcet. Le vainqueur de l'élection avec la méthode de Borda est l'alternative qui reçoit le plus de points de la part des agents, chaque agent donnant autant de points à une alternative que le nombre d'alternatives qu'il considère plus mauvaises. Par conséquent, lorsque les préférences des agents sont imprécisément connues, la mise en œuvre de l'approche incrémentale avec la méthode de Borda nécessite de résoudre des problèmes d'optimisation tenant compte de tous les classements possibles des alternatives. Pour résoudre ces problèmes de manière efficace, nous avons introduit une méthode de calcul exacte utilisant la programmation linéaire en nombres entiers pour pouvoir considérer tous les classements admissibles de manière implicite et compacte. Dans nos expériences, nous avons constaté que la relaxation linéaire de ces programmes ne dégrade pas la précision des calculs outre mesure tout en permettant une résolution en temps polynomial.

Domaine combinatoire. Pour des raisons d'efficacité, l'élicitation incrémentale fondée sur le concept de regret est généralement réservée aux problèmes de décision avec un faible nombre d'alternatives. Comme deuxième source de complexité, nous avons donc envisagé des situations de décision dans lesquelles les alternatives possibles sont en nombre combinatoire et définies de manière implicite. Dans ce contexte, nous avons proposé et étudié une nouvelle approche consistant à entremêler l'élicitation incrémentale et la résolution du problème combinatoire, pour tenter de déterminer le plus rapidement possible la meilleure alternative. Pour mettre en œuvre cette approche, nous avons tout d'abord introduit des algorithmes permettant de trouver toutes les alternatives *potentiellement optimales* : une alternative est potentiellement optimale si et seulement si elle est optimale pour au moins un profil de préférences compatible avec les informations que nous avons collectées sur les préférences des agents. Nous avons ensuite proposé de poser des questions à des moments clés de la résolution de manière à garantir que ces algorithmes retournent à la fin une alternative *nécessairement optimale*, c'est-à-dire une solution qui est optimale pour tous les profils de préférences compatibles avec les données préférentielles recueillies. Pour ne pas cumuler les difficultés, nous avons étudié en premier lieu des problèmes de décision dans lesquels les préférences des agents sont représentables par des modèles décisionnels simples (e.g., somme pondérée, utilité espérée). En décision multicritère, nous avons étudié le problème de recherche dans un graphe d'états (cf. Section 3.1) ainsi que le problème d'arbres couvrants (cf. Section 3.2), en supposant que les préférences du décideur sont représentables par une somme pondérée. Pour ces problèmes, nous avons élaboré des algorithmes interactifs utilisant respectivement la programmation dynamique et le principe glouton afin de déterminer la meilleure alternative pour le décideur. Comme problème de vote sur domaine combinatoire, nous avons considéré le problème de sac à dos multi-agents avec vote par approbation (cf. Section 4.2). En supposant que les préférences des agents sur les sacs à dos possibles sont additives, nous avons proposé un algorithme interactif par séparation et évaluation permettant de déterminer le sac à dos approuvé par le plus grand nombre de personnes. Dans le cadre de la décision dans le risque, nous nous sommes intéressés aux problèmes de décision séquentielle se modélisant sous la forme d'arbre de décision (cf. Chapitre 5). Plus précisément, nous avons conçu un algorithme interactif par induction arrière permettant de construire une stratégie optimale au sens de l'utilité sociale espérée.

Tous ces algorithmes ont été testés expérimentalement et nous avons observé que ces derniers sont plus efficaces que les approches classiques ne combinant pas la recherche et l'élicitation incrémentale.

Modélisation complexe sur domaine combinatoire. Enfin, nous avons abordé le problème de recherche dans un graphe d'états multi-objectifs avec une utilité espérée à la Choquet (cf. Section 3.1.5). Il s'agissait pour nous de tenter de surmonter à la fois la difficulté liée à une modélisation sophistiquée et celle engendrée par la nature combinatoire de l'espace des solutions. L'utilisation d'un modèle non linéaire (comme l'utilité espérée à la Choquet) sur domaine combinatoire exclut la possibilité d'exploiter le modèle de préférence pour éliminer en cours de résolution des sous-solutions par comparaison avec d'autres. Pour compenser cette perte des coupes locales due à l'utilisation d'un modèle décisionnel non linéaire, nous avons proposé une procédure interactive inspirée de la littérature sur l'optimisation combinatoire approchée pour réduire la taille de l'arbre d'exploration. Pour pouvoir rétablir notre garantie de performance (en terme de regrets), nous avons introduit un nouveau type de regret, que nous avons accompagné d'une stratégie de génération de questions permettant de garantir la convergence de notre procédure de recherche interactive. Expérimentalement, nous avons observé le caractère opérationnel de notre procédure sur des instances de taille relativement grande. Cette observation nous encourage à poursuivre dans cette voie pour aborder d'autres problèmes de décision sur domaine combinatoire en présence de préférences complexes.

Recherches futures

Les travaux menés durant cette thèse laissent entrevoir de nombreuses perspectives de recherche intéressantes pour la problématique de résolution par élicitation incrémentale des préférences. Nous présentons maintenant celles qui nous semblent devoir être explorées en priorité.

Borner le nombre de questions. Tout au long de cette thèse, nous avons constaté que l'approche d'élicitation incrémentale permet en pratique de trouver rapidement la meilleure alternative possible, en posant relativement peu de questions aux agents décisionnels. Toutefois, selon la stratégie de génération de questions envisagée et le problème de décision considéré, il n'est pas toujours évident de déterminer une borne supérieure théorique sur le nombre de questions produites par une procédure de résolution par élicitation incrémentale. Pour les problèmes d'arbres couvrants multicritères et de décision séquentielle dans les arbres de décision, nous sommes parvenus à proposer des algorithmes interactifs de complexité polynomiale, posant un nombre polynomial de questions dans le pire cas. En revanche, nous ne sommes pas arrivés à majorer finement le nombre de questions que peuvent produire nos algorithmes interactifs conçus pour le problèmes de recherche dans un graphe d'états multi-objectifs et de sac à dos avec vote par approbation. À première vue, pour pouvoir détecter la meilleure solution, ces algorithmes interactifs pourraient avoir besoin de poser autant de questions par agent que le nombre de solutions réalisables. Pour nous en convaincre, nous pourrions aborder ces problèmes d'élicitation sous l'angle de la *complexité de communication* (e.g., [Kushilevitz and Nisan, 1997, Conitzer and Sandholm, 2002, 2005,

Nisan and Segal, 2006, Procaccia and Rosenschein, 2006a, Brandt et al., 2013]). Par exemple, nous pourrions essayer de déterminer une borne inférieure sur le nombre de questions nécessaires à l'identification de la meilleure alternative, indépendamment de la procédure d'élicitation utilisée.

Compatibilité du modèle et gestion des incohérences. Il est important de souligner que l'absence de redondances dans les questions engendrées par nos algorithmes est un élément clé dans la production de questionnaires efficaces. En effet, nous posons une question aux agents uniquement si toutes les réponses possibles sont compatibles avec les informations préférentielles que nous avons déjà collectées. Cet avantage pratique force de manière implicite la cohérence des données recueillies et des préférences que nous pouvons inférer. De ce fait, nos procédures interactives ne contrôlent ni la cohérence interne des agents ni l'adéquation du modèle décisionnel avec leurs préférences, ce dernier problème étant supposé réglé préalablement. Revisiter l'approche incrémentale pour tenir compte des incohérences éventuelles et autoriser la falsification du modèle, tout en préservant l'efficacité pratique de cette approche, constitue une ligne de recherche très intéressante. Nous pourrions commencer par essayer d'exploiter les travaux existants sur la *programmation par contraintes souples*, qui étudie des problèmes combinatoires dont les solutions possibles sont définies implicitement par des contraintes pouvant être transgressées (e.g., [Fargier and Lang, 1993, Schiex et al., 1995, Rossi et al., 2006, 2009]). La programmation par contraintes souples permet en particulier de travailler avec des systèmes de contraintes n'admettant aucune solution réalisable. De ce fait, pour pouvoir tester la cohérence des agents, nous pourrions envisager de leur poser des questions dont la réponse peut être déduite des informations disponibles et de gérer les éventuelles incohérences avec les outils développés pour la programmation par contraintes souples. Pour la remise en cause du modèle décisionnel, nous pourrions par exemple décider d'interrompre le processus d'élicitation lorsque le système de contraintes contient trop d'incohérences. À ce moment là, il conviendrait de remplacer le modèle considéré par un autre modèle décisionnel satisfaisant un plus grand nombre de contraintes imposées par les données de préférences collectées.

Autres modèles de préférences. Une extension directe de nos travaux consiste à proposer des méthodes de résolution par élicitation incrémentale pour d'autres modèles de préférences (e.g., HWA [Llamazares, 2011], WOWA [Torra, 1997], RDU [Quiggin, 1992]...). En particulier, le modèle RDU permet d'étendre les possibilités descriptives offertes par le modèle EU en considérant par ailleurs une fonction de déformation des probabilités. Par conséquent, il serait plaisant de parvenir à étendre au modèle RDU nos travaux menés sur les arbres de décision avec le modèle EU. La principale difficulté réside dans le fait que le critère RDU n'est pas cohérent dynamiquement (contrairement au modèle EU). De ce fait, il paraît difficile de construire une stratégie RDU-optimale à l'aide d'un algorithme par induction arrière (même lorsque le modèle de préférence est complètement caractérisé). Cette observation a conduit à la proposition d'un algorithme par séparation et évaluation pour la détermination d'une stratégie RDU-optimale au sens du critère *minimax regret* avec modèle précis [Jeantet et al., 2012]. Cet algorithme pourrait servir de point de départ pour la conception d'une procédure de décision par élicitation incrémentale pour le modèle RDU dans les arbres de décision. Par ailleurs, un algorithme

interactif a récemment été proposé pour éliciter le modèle RDU de manière incrémentale sur domaine non combinatoire [Perny et al., 2016]. Nous pourrions essayer de combiner ces deux derniers algorithmes pour produire une procédure de recherche par élicitation incrémentale pour le modèle RDU dans les arbres de décision.

Autres problèmes d'optimisation combinatoire. Une prolongation naturelle des travaux réalisés durant cette thèse est de considérer d'autres problèmes d'optimisation combinatoire (e.g., problèmes multicritères de flots, de voyageur de commerce, d'affectation, d'emplacement d'installations...). En particulier, nous pourrions envisager la problématique d'allocation de ressources avec vote par approbation. Il s'agirait de déterminer la répartition des ressources sur les différents agents qui est approuvée par le plus grand nombre de personnes. Le problème d'allocation de ressources est un problème d'optimisation combinatoire étudié à l'origine en économie et faisant actuellement partie des sujets phares en choix social computationnel. Pour concevoir un algorithme de résolution par élicitation incrémentale pour ce problème de vote, nous pourrions donc nous inspirer des méthodes proposées dans la littérature pour résoudre des problèmes d'allocation équitable avec préférences parfaitement connues (e.g., [Brandt et al., 2016] partie II). Notons par ailleurs que cette problématique est relativement proche du problème de sac à dos multi-agents que nous avons introduit en Section 4.2, qui consiste à déterminer le sous-ensemble d'objets approuvé par le plus de personnes. Il serait intéressant de savoir si notre procédure interactive par séparation et évaluation spécialement conçue pour le problème de sac à dos multi-agents peut être étendue efficacement au problème d'allocation de ressources.

Manipulation des procédures de décision collective. Dans cette thèse, nous avons introduit plusieurs procédures de recherche par élicitation incrémentale pour résoudre des problèmes de décision collective. Sous leur forme actuelle, ces procédures ne se protègent pas contre les éventuelles manipulations orchestrées par des agents pour faire élire une alternative qu'ils estiment meilleure. Signalons que la question de la manipulation des systèmes de vote est un sujet très actif dans le domaine du choix social (e.g., [Brandt et al., 2016] chapitre 6). Au sein de nos procédures, les manipulations possibles sont sous la forme de mauvaises réponses transmises volontairement au système par des votants dans le but de maximiser leur propre profit. Pour nous prémunir contre ce type de comportement relativement "égoïste", nous pourrions par exemple essayer de modifier nos procédures en nous inspirant des mécanismes de type Vickrey-Clarke-Groves (e.g., [Vickrey, 1961, Clarke, 1971, Groves, 1973]) qui sont des mécanismes à véracité garantie permettant de déterminer une alternative optimale pour le critère utilitaire en incluant un système de paiements monétaires. Combiner ce type de mécanisme avec l'approche d'élicitation incrémentale semble constituer une piste de recherche prometteuse pour la décision collective sur domaine combinatoire.

Annexe

Démonstration de la proposition 9 (chapitre 2). Il s'agit de prouver que l'algorithme 1 construit une solution optimale du programme LP_2. Pour ce faire, il suffit de montrer par récurrence que la proposition suivante, notée $P(j)$, est vraie à la fin de chaque itération $j \in \{0, \ldots, |A^-|\}$:

$P(j)$: l'instanciation des variables dans $\{v_{A_l^-} : 1 \leq l \leq j\}$, notée I_j, peut être étendue en une solution optimale du programme LP_2.

En effet, si la proposition $P(j)$ est vraie pour $j = |A^-|$, alors nous savons que l'instanciation des variables dans $\{v_A : A \in A^-\}$ peut être étendue en une instanciation complète optimale. Par ailleurs, nous savons aussi que les variables dans $\{v_A : A \in A^+\}$ vérifient les conditions nécessaires d'optimalité à la fin de l'algorithme (cf. proposition 8). Par conséquent, nous pouvons conclure que l'instanciation des variables dans $\{v_A : A \in A^-\} \cup \{v_A : A \in A^+\}$ forme nécessairement une instanciation complète optimale.

Pour l'itération $j = 0$ (avant d'entrer dans la boucle "for"), la proposition $P(j)$ est trivialement vraie puisque l'ensemble $\{v_{A_l^-} : 1 \leq l \leq j\}$ est vide. Supposons maintenant que la proposition $P(j-1)$ est vraie pour une certaine itération $j \in \{1, \ldots, |A^-|\}$. Montrons alors que la proposition $P(j)$ est aussi vraie. Tout d'abord, notons que l'algorithme ne modifie aucune des variables de l'ensemble $\{v_{A_l^-} : 1 \leq l \leq j-1\}$ durant l'itération j ; ainsi, l'instanciation I_j ne diffère de l'instanciation I_{j-1} que sur la variable $v_{A_j^-}$. De plus, par hypothèse de récurrence, il existe au moins une instanciation complète optimale étendant l'instanciation partielle I_{j-1}. Par conséquent, pour montrer $P(j)$, il suffit de prouver que, parmi les instanciations complètes optimales étendant l'instanciation partielle I_{j-1}, il en existe au moins une qui étend aussi l'instanciation I_j. C'est ce à quoi nous nous attelons maintenant.

La valeur de la variable $v_{A_j^-}$ est déterminée durant la $j^{\text{ème}}$ itération en exécutant une boucle "while". Deux scénarios sont alors possibles : la valeur de $v_{A_j^-}$ est fixée à la ligne 4 ou à la ligne 12. Pour simplifier la preuve, nous supposons maintenant que la valeur de $v_{A_j^-}$ est déterminée par la ligne 12 (nous reviendrons sur cette hypothèse à la fin de cette preuve). Plus précisément, nous supposons que la boucle "while" s'arrête à une certaine itération $k \in \{1, \ldots, |D^j|\}$ à cause du test en ligne 11 et que la valeur u_k^j est affectée à la variable $v_{A_j^-}$ en ligne 12, où u_k^j est la valeur de la variable $v_{D_k^j}$ au début de l'itération j. D'après la proposition 7, nous avons :

$$u_k^j = \min\{u_{D_k^j}, \min_{A \in Pa_{j-1}(D_k^j)} v_A\}$$

où $Pa_{j-1}(D_k^j) = \{A_l^-, 1 \leq l \leq j-1 : D_k^j \subset A_l^-\}$. Cette égalité nous informe que la valeur u_k^j représente la plus grande valeur admissible pour la variable $v_{D_k^j}$ sachant les valeurs prises par les variables dans $\{v_A : A \in Pa_{j-1}(D_k^j)\}$. Comme en plus nous avons $\{v_A : A \in Pa_{j-1}(D_k^j)\} \subseteq \{v_{A_l^-} : 1 \leq l \leq j-1\}$, alors nous en déduisons qu'il existe au moins une instanciation complète optimale qui étend I_{j-1} tout en vérifiant $v_{D_k^j} \leq u_k^j$. De ce fait, nous pouvons ajouter la contrainte $v_{D_k^j} \leq u_k^j$ au programme LP_2 tout en restant capable de trouver une solution optimale au programme LP_2. Ainsi, nous étudions à présent le problème d'optimisation suivant :

$$\max_{u \in [l_{D_k^j}, u_k^j]} f(u, \mathcal{A}_{(x,y)}) \tag{5.12}$$

où $f(u, \mathcal{A}_{(x,y)})$ représente la valeur optimale du sous-problème suivant :

$$\begin{aligned} \max_{v_A, A \in \mathcal{A}_{(x,y)}} \quad & \sum_{A \in \mathcal{A}_{(x,y)}} c_A v_A \\ \text{s.c.} \quad & v_{D_k^j} = u \\ & \text{Équations (2.28-2.32)} \end{aligned} \tag{5.13}$$

À présent, notre objectif est de montrer que le problème d'optimisation (5.12) admet une solution optimale qui étend l'instanciation I_{j-1} tout en vérifiant $v_{A_j^-} = u_k^j$. Pour y parvenir, nous allons montrer que la fonction $f(u, \mathcal{A}_{(x,y)})$ atteint sa valeur maximale en $u = u_k^j$. Soit $u \in [l_{D_k^j}, u_k^j]$ une valeur quelconque. Afin de calculer la valeur $f(u, \mathcal{A}_{(x,y)})$, nous introduisons les ensembles suivants :

- $S(D_k^j) = \{A \in \mathcal{A}_{(x,y)} : A \supseteq D_k^j\}$: l'ensemble des sur-ensembles de D_k^j restreint aux ensembles de $\mathcal{A}_{(x,y)}$. Chaque élément de $S(D_k^j)$ est associé à une variable qui est bornée inférieurement par la valeur u (à cause de la contrainte $v_{D_k^j} = u$).
- $P(D_k^j) = \{A \in \mathcal{A}_{(x,y)} : A \subset D_k^j\}$: l'ensemble des parties de D_k^j restreint aux ensembles de $\mathcal{A}_{(x,y)}$. Chaque élément de l'ensemble $P(D_k^j)$ correspond à une variable qui est bornée supérieurement par u (encore une fois à cause de la contrainte $v_{D_k^j} = u$).
- $I(D_k^j) = \{A \in \mathcal{A}_{(x,y)} : A \not\supseteq D_k^j, A \not\subset D_k^j\}$: les ensembles de $A \in \mathcal{A}_{(x,y)}$ qui sont incomparables avec l'ensemble D_k^j pour l'inclusion. Notons que nous avons nécessairement $I(D_k^j) \cap A^+ = \emptyset$ puisque $D_k^j \in A^+$ et que A^+ est une séquence emboîtée. Par conséquent, tous les éléments de $I(D_k^j)$ font en réalité partie de la séquence A^-. Par définition de la séquence A^-, nous avons $c_A < 0$ pour tout $A \in I(D_k^j)$; ainsi, la fonction objectif du problème (5.13) augmente strictement lorsque la variable v_A diminue. Montrons que les solutions optimales du problème (5.13) vérifient $v_A \leq \max\{l_A, u\}$ pour tout $A \in I(D_k^j)$. Soit $A \in I(D_k^j)$. Tout d'abord, supposons qu'il n'existe aucun ensemble $A' \in \mathcal{A}_{(x,y)}$ tel que $A' \subset A$. Dans ce cas, les équations (2.28-2.30) n'induisent aucune contrainte de type $v_{A'} \leq v_A$ avec $A' \in \mathcal{A}_{(x,y)}$. Comme en plus la fonction objectif du problème (5.13) augmente strictement lorsque la variable v_A diminue, alors nous en déduisons que les solutions optimales de ce problème vérifient $v_A = l_A \leq \max\{l_A,\ u\}$. Supposons maintenant qu'il existe un ensemble $A' \in \mathcal{A}_{(x,y)}$ tel que $A' \subset A$. Nous allons commencer par montrer que la contrainte

$v_A \leq \max\{l_A, u\}$ n'impacte aucunement la variable $v_{A'}$. Deux cas sont à distinguer : $c_{A'} > 0$ ou $c_{A'} < 0$. Dans le premier cas, notons que l'inclusion $A' \subset D_k^j$ est nécessairement vraie car sinon nous aurions $D_k^j \subseteq A' \subset A$ (ce qui contredirait le fait que A appartienne à $I(D_k^j)$). Par conséquent, A' appartient forcément à l'ensemble $P(D_k^j)$ et donc $v_{A'}$ est bornée supérieurement par la valeur u. De ce fait, imposer la contrainte $v_A \leq \max\{l_A, u\}$ n'impacte aucunement la variable $v_{A'}$. Plaçons-nous à présent dans le deuxième cas : $A' \subset A$ et $c_{A'} < 0$. Notons que nous avons soit $A' \in P(D_k^j)$ soit $A' \in I(D_k^j)$ car sinon nous aurions $D_k^j \subseteq A' \subset A$ (ce qui contredirait le fait que A appartienne à $I(D_k^j)$). Si $A' \in P(D_k^j)$, alors $v_{A'}$ est bornée supérieurement par la valeur u et donc la contrainte $v_A \leq \max\{l_A, u\}$ n'a aucun impact sur la variable $v_{A'}$. Si en revanche $A' \in I(D_k^j)$, alors il s'agit d'itérer tout le raisonnement sur l'ensemble A' pour obtenir le résultat. Ainsi, dans tous les cas, nous pouvons imposer la contrainte $v_A \leq \max\{l_A, u\}$ sans impacter la variable $v_{D_k^j}$. Ce résultat nous permet de déduire que toutes les solutions optimales du problème (5.13) vérifient $v_A \leq \max\{l_A, u\}$ car la fonction objectif de ce problème augmente strictement lorsque v_A diminue.

Finalement, nous venons de décomposer l'ensemble $\mathcal{A}_{(x,y)}$ en trois sous-ensembles disjoints $S(D_k^j)$, $P(D_k^j)$ et $I(D_k^j)$ tels que :

$$f(u, \mathcal{A}_{(x,y)}) = h(u, P(D_k^j), I(D_k^j)) + g(u, S(D_k^j))$$

où $h(u, P(D_k^j), I(D_k^j))$ représente la valeur optimale du problème d'optimisation suivant :

$$\begin{aligned} \max_{v_A, A \in P(D_k^j) \cup I(D_k^j)} \quad & \sum_{A \in P(D_k^j) \cup I(D_k^j)} c_A v_A \\ \text{s.c.} \quad & v_A \leq u, \, \forall A \in P(D_k^j) \\ & v_A \leq \max\{l_A, u\}, \, \forall A \in I(D_k^j) \\ & \text{Équations (2.28-2.32) restreintes à } P(D_k^j) \cup I(D_k^j) \end{aligned}$$

et $g(u, S(D_k^j))$ représente celle du problème suivant :

$$\begin{aligned} \max_{v_A, A \in S(D_k^j)} \quad & \sum_{A \in S(D_k^j)} c_A v_A \\ \text{s.c.} \quad & v_{D_k^j} = u \\ & v_A \geq u, \, \forall A \in S(D_k^j) \\ & \text{Équations (2.28-2.32) restreintes à } S(D_k^j) \end{aligned}$$

Observons que $h(u, P(D_k^j), I(D_k^j))$ est une fonction qui croît avec la valeur u puisque les contraintes de type "$v_A \leq u$" ou "$v_A \leq \max\{l_A, \, u\}$" deviennent de moins en moins contraignantes lorsque la valeur u augmente. Cette observation nous permet de conclure que la fonction $h(u, P(D_k^j), I(D_k^j))$ est maximale

pour la valeur $u = u_k^j$. Ainsi, pour montrer que la fonction $f(u, \mathcal{A}_{(x,y)})$ atteint sa valeur maximale en $u = u_k^j$, il suffit de prouver que la fonction $g(u, S(D_k^j))$ est maximale pour $u = u_k^j$.

Commençons par simplifier le calcul de la valeur $g(u, S(D_k^j))$. Rappelons que, par hypothèse de récurrence, toutes les variables $\{v_{A_l^-} : 1 \leq l \leq j-1\}$ ont été "correctement" instanciées. Par conséquent, ces variables peuvent être retirées du problème d'optimisation ; il suffit en effet de contraindre les futures instanciations à être compatibles avec leur valeur. De plus, étant donnée l'instanciation de ces variables, nous pouvons obtenir la valeur optimale de chaque variable de l'ensemble $\{v_A, A \in A^+ : A \not\subset A_j^-\}$ en utilisant les conditions nécessaires d'optimalité décrites par l'équation (2.26) ; ces variables peuvent donc elles aussi être supprimées du problème d'optimisation. Finalement, nous souhaitons montrer que la fonction $g(u, s(D_k^j))$ est maximale pour $u = u_k^j$ où :

$$s(D_k^j) = S(D_k^j) \backslash (\{A_l^- : 1 \leq l \leq j-1\} \cup \{A \in A^+ : A \not\subset A_j^-\})$$

Soit $u \in [l_{D_k^j}, u_k^j]$. Supposons tout d'abord que u est une valeur réalisable pour toutes les variables dans $\{v_A : A \in s(D_k^j)\}$; autrement dit, nous supposons que $l_A \leq u \leq u_A$ pour tout $A \in s(D_k^j)$. Sous cette hypothèse, nous allons montrer qu'il existe une valeur $\alpha > 0$ telle que $g(u, s(D_k^j)) = \alpha \times u$. Pour ce faire, nous commençons par décomposer l'ensemble $s(D_k^j)$ en deux sous-ensembles $s(D_k^j)^+$ et $s(D_k^j)^-$:

- $s(D_k^j)^+ = A^+ \cap s(D_k^j) = \{D_m^j : 1 \leq m \leq k\}$. Cet ensemble est composé des k premiers descendants positifs du nœud A_j^-.
- $s(D_k^j)^- = A^- \cap s(D_k^j) = \{A_l^- : j \leq l \leq L\}$, où L est la plus grande valeur l telle que $j \leq l \leq |A^-|$ et $A_l^- \supset D_k^j$. Cet ensemble est composé de tous les ensembles $A \in A^-$ tels que $D_k^j \subset A \subseteq A_j^-$.

Montrons tout d'abord que toutes les variables de l'ensemble $\{v_A : A \in s(D_k^j)^-\}$ peuvent être retirées du problème d'optimisation. Pour tout $l \in \{k, \ldots, L\}$, comme $D_k^j \subset A_l^- \subseteq A_j^-$, alors D_1^l appartient nécessairement à l'ensemble $s(D_k^j)^+$. Par conséquent, nous pouvons décomposer $s(D_k^j)^-$ en k sous-ensembles $s(D_k^j)_m^-$, $m \in \{1, \ldots, k\}$, définis comme suit :

$$s(D_k^j)_m^- = \{A_l^-, j \leq l \leq L : D_1^l = D_m^j\}$$

Soit $A \in s(D_k^j)_m^-$. D'une part, comme $c_A < 0$, alors $g(u, s(D_k^j))$ augmente lorsque v_A diminue. D'autre part, comme $D_m^j = D_1^l$ par définition de $s(D_k^j)_m^-$, alors nous avons :

$$D_m^j \subset A \text{ et } \nexists A' \in A^+, D_m^j \subset A' \subset A$$

Par conséquent, les contraintes de monotonie imposées par les équations (2.28-2.30) induisent l'égalité $v_A = \max\{l_A, v_{D_m^j}\}$ à l'optimum. Comme par ailleurs nous avons $v_{D_m^j} \geq u$ (par définition de g) et $u \geq l_A$ (par hypothèse), alors nous avons en réalité $v_A = v_{D_m^j}$ à l'optimum. Ceci nous permet de substituer la variable v_A par la variable $v_{D_m^j}$ dans le problème d'optimisation. De ce fait, la valeur $g(u, s(D_k^j))$ peut

être calculée en résolvant le problème d'optimisation suivant :

$$\max_{v_{D_m^j}, m\in\{1,\dots,k\}} \sum_{m=1}^{k} \Big(c_{D_m^j} + \sum_{A\in s(D_k^j)_m^-} c_A\Big) v_{D_m^j} \tag{5.14}$$

$$\text{s.c.} \quad v_{D_k^j} = u$$

$$v_{D_m^j} \geq u,\ \forall m \in \{1,\dots,k-1\}$$

$$\text{Équations (2.28-2.32) restreintes à } s(D_k^j)^+$$

Observons que ce problème d'optimisation n'implique que les variables positives $v_{D_m^j}, m \in \{1,\dots,k\}$. Pour montrer qu'il existe une valeur $\alpha > 0$ telle que $g(u, s(D_k^j)) = \alpha \times u$, nous allons prouvé que le problème (5.14) admet une solution optimale telle que $v_{D_1^j} = \dots = v_{D_k^j}$. Pour ce faire, nous allons tout d'abord étudier le terme associé à la variable $v_{D_1^j}$ dans la fonction objectif du problème (5.14). Le coefficient de la variable $v_{D_1^j}$ dans la fonction objectif est en réalité égal à $c^+ + c^-$, où c^+ et c^- sont deux valeurs définies à la première itération de la boucle "while". Puisque la boucle "while" s'arrête à l'itération k à cause du test en ligne 11, alors l'inégalité $c^+ + c^- < 0$ est forcément fausse à la première itération (sauf si $k = 1$ bien sûr) ; par conséquent, la fonction objectif (5.14) augmente lorsque $v_{D_1^j}$ diminue. Par ailleurs, les contraintes de monotonie imposent uniquement la contrainte $v_{D_1^j} \geq v_{D_2^j}$. En conséquence, le problème (5.14) admet une solution optimale telle que $v_{D_1^j} = \max\{l_{D_1^j}, v_{D_2^j}\}$. Comme en plus nous avons $v_{D_2^j} \geq u$ (par définition de g) et $u \geq l_{D_1^j}$ (par hypothèse), alors nous en concluons que cette solution vérifie $v_{D_1^j} = v_{D_2^j}$. Ce résultat nous permet de substituer la variable $v_{D_1^j}$ par la variable $v_{D_2^j}$, ce qui nous conduit au problème d'optimisation suivant :

$$\max_{v_{D_m^j}, m\in\{2,\dots,k\}} \Big[\sum_{m=1}^{2} \Big(c_{D_m^j} + \sum_{A\in s(D_k^j)_m^-} c_A\Big)\Big] v_{D_2^j} + \sum_{m=3}^{k} \Big(c_{D_m^j} + \sum_{A\in s(D_k^j)_m^-} c_A\Big) v_{D_m^j}$$

$$\text{s.c.} \quad v_{D_k^j} = u$$

$$v_{D_m^j} \geq u,\ \forall m \in \{2,\dots,k-1\}$$

$$\text{Équations (2.28-2.32) restreintes à } s(D_k^j)^+ \setminus \{D_1^j\}$$

En itérant le raisonnement, nous pouvons conclure que le problème (5.14) admet une solution optimale telle que $v_{D_1^j} = \dots = v_{D_k^j}$ et que la valeur $g(u, s(D_k^j))$ peut être obtenue en résolvant le problème suivant :

$$\max_{v_{D_k^j}} \Big(\sum_{A\in s(D_k^j)} c_A\Big) v_{D_k^j}$$

$$\text{s.c.} \quad v_{D_k^j} = u$$

De ce fait, nous avons finalement :

$$g(u, s(D_k^j)) = \Big(\sum_{A \in s(D_k^j)} c_A \Big) \times u = \alpha \times u$$

avec $\alpha = \sum_{A \in s(D_k^j)} c_A$. Puisque la boucle "while" s'arrête à l'itération k à cause du test en ligne 11, alors nous avons $c^+ + c^- > 0$ à cette itération ; ainsi, nous avons $\alpha = c^+ + c^- > 0$.

Soulignons le fait que l'égalité $g(u, s(D_k^j)) = \alpha \times u$ est vraie seulement si $u \geq l_A$ pour tout $A \in s(D_k^j)$. Dans le cas contraire, certaines variables $v_A, A \in s(D_k^j)$, ont été autorisées à décroître en dessous de leur borne inférieure l_A durant le raisonnement (pour améliorer la fonction objectif) et donc nous avons uniquement : $g(u, s(D_k^j)) \leq \alpha \times u$. Cependant, cette égalité est bien vraie pour $u = u_k^j$. En effet, d'une part l'inégalité $u_k^j \geq l_A$ est vraie pour tout $A \in s(D_k^j)$ puisque $u_k^j \geq l_{A_j^-}$ (cf. ligne 8) et d'autre part l'inégalité $l_{A_j^-} \geq l_A$ est vraie pour tout $A \in s(D_k^j)$ puisque $A_j^- \supset A$. Par conséquent, pour tout $u \neq u_k^j$, nous avons :

$$g(u, s(D_k^j)) \leq \alpha \times u < \alpha \times u_k^j = g(u_k^j, s(D_k^j))$$

Ceci montre que la fonction $g(u, s(D_k^j))$ atteint sa valeur maximale en $u = u_k^j$; ainsi, la fonction $f(u, \mathcal{A}_{(x,y)})$ atteint aussi sa valeur maximale pour $u = u_k^j$. En résumé, pour le cas où $v_{A_j^-}$ est fixée à la valeur u_k^j (à cause de la condition en ligne 11), nous venons de montrer que le programme LP_2 admet une solution optimale qui étend l'instanciation I_{j-1} tout en vérifiant $v_{A_j^-} = v_{D_1^j} = v_{D_k^j} = u_k^j$; ainsi, la proposition $P(j)$ est vrai dans ce cas là.

Pour le cas où $v_{A_j^-}$ est fixée à la valeur $l_{A_j^-}$ (cf. ligne 4), deux scénarios sont à distinguer :

- Si $D^j = \emptyset$ ou $v_{D_1^j} \leq l_{A_j^-}$, alors aucune variable positive n'est "en conflit" avec la variable $v_{A_j^-}$. Comme par ailleurs la fonction objectif augmente strictement lorsque $v_{A_j^-}$ décroît, alors nous savons que $v_{A_j^-}$ doit être fixée à sa borne inférieure $l_{A_j^-}$ dans ce cas.
- Dans le cas contraire, nous pouvons reprendre les mêmes arguments que ceux utilisés pour le cas où $v_{A_j^-}$ est fixée à la valeur u_k^j (en ligne 12). Plus précisément, un raisonnement similaire avec $k = M$ permet d'établir le résultat, où M est la plus grande valeur m telle que $1 \leq m \leq |D^j|$ et $v_{D_m^j} > l_{A_j^-}$. La principale différence réside dans le fait que nous obtenons une fonction qui décroît en fonction de u à la fin du raisonnement (puisque tous les tests en ligne 11 échouent) et donc la variable $v_{A_j^-}$ doit être mise à sa borne inférieure $l_{A_j^-}$ à la place.

Ainsi, la proposition $P(j)$ est vérifiée dans tous les cas, ce qui signifie que notre algorithme est valide. □

Bibliographie

James M Abello and Charles R Johnson. How large are transitive simple majority domains ? *SIAM Journal on Algebraic Discrete Methods*, 5(4) :603–618, 1984. *Cité page 32.*

Julien Ah-Pine, Brice Mayag, and Antoine Rolland. Identification of a 2-additive bi-capacity by using mathematical programming. In *Algorithmic Decision Theory*, pages 15–29. Springer, 2013. *Cité page 119.*

Nir Ailon. An active learning algorithm for ranking from pairwise preferences with an almost optimal query complexity. *Journal of Machine Learning Research*, 13(Jan) :137–164, 2012. *Cité page 91.*

Maurice Allais. La comportement de l'homme rationnel devant le risque : critique des postulats de l'école américaine. *Econometrica*, 21 :503–546, 1953. *Cité page 47.*

Silvia Angilella, Salvatore Greco, and Benedetto Matarazzo. Non-additive robust ordinal regression : A multiple criteria decision model based on the Choquet integral. *European Journal of Operational Research*, 201(1) :277–288, 2010. *Cité page 53.*

Kenneth J. Arrow. *Social choice and individual values.* New Haven, CT : Cowles Foundation, 1951. *Cité page 35.*

Haris Aziz, Markus Brill, Vincent Conitzer, Edith Elkind, Rupert Freeman, and Toby Walsh. Justified representation in approval-based committee voting. In *Proceedings of AAAI'15*, pages 784–790, 2015a. *Cité page 193.*

Haris Aziz, Markus Brill, Felix A. Fischer, Paul Harrenstein, Jérôme Lang, and Hans Georg Seedig. Possible and necessary winners of partial tournaments. *J. Artif. Intell. Res. (JAIR)*, 54 :493–534, 2015b. *Cité page 122.*

Haris Aziz, Serge Gaspers, Joachim Gudmundsson, Simon Mackenzie, Nicholas Mattei, and Toby Walsh. Computational aspects of multi-winner approval voting. In *Proceedings of AAMAS'15*, pages 107–115, 2015c. *Cité page 193.*

Carlos A. Bana e Costa and Jean-Claude Vansnick. MACBETH-an interactive path towards the construction of cardinal value functions. *International transactions in operational Research*, 1(4) :489–500, 1994. *Cité page 64.*

Carlos A. Bana e Costa and Jean-Claude Vansnick. Cardinal value measurement with MACBETH. In S.H. Zanakis, G. Doukidis, and C. Zopounidis, editors, *Decision Making : Recent Developments and Worldwide Applications*, pages 317–329. Kluwer, 2000. *Cité pages 64 et 141.*

Antoinette Baujard, Herrade Igersheim, Isabelle Lebon, Frédéric Gavrel, and Jean-François Laslier. Who's favored by evaluative voting ? an experiment conducted during the 2012 french presidential election. *Electoral Studies*, 34 :131–145, 2014. *Cité page 39.*

John C Beatty and Brian A Barsky. *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann, 1995. *Cité page 234.*

Richard Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 6, 1957. *Cité page 214.*

Nawal Benabbou and Patrice Perny. On possibly optimal tradeoffs in multicriteria spanning tree problems. In *Proceedings of ADT'15*, pages 322–337, 2015a. *Cité page 121.*

Nawal Benabbou and Patrice Perny. Incremental weight elicitation for multiobjective state space search. In *Proceedings of AAAI'15*, pages 1093–1098, 2015b. *Cité page 121.*

Nawal Benabbou and Patrice Perny. Combining preference elicitation and search in multiobjective state-space graphs. In *Proceedings of IJCAI'15*, pages 297–303, 2015c. *Cité page 121.*

Nawal Benabbou and Patrice Perny. Solving multi-agent knapsack problems using incremental approval voting. In *Proceedings of ECAI'16*, pages 1318–1326, 2016. *Cité page 169.*

Nawal Benabbou and Patrice Perny. Adaptive elicitation of preferences under uncertainty in sequential decision making problems. In *Proceedings of IJCAI'17*, 2017. *Cité page 213.*

Nawal Benabbou, Patrice Perny, and Paolo Viappiani. Incremental elicitation of Choquet capacities for multicriteria decision making. In *Proceedings of ECAI'14*, pages 87–92, 2014. *Cité page 63.*

Nawal Benabbou, Christophe Gonzales, Patrice Perny, and Paolo Viappiani. Minimax regret approaches for preference elicitation with rank-dependent aggregators. *EURO Journal on Decision Processes*, 3 (1) :29–64, 2015. *Cité pages 59 et 63.*

Nawal Benabbou, Patrice Perny, and Paolo Viappiani. A regret-based preference elicitation approach for sorting with multicriteria reference profiles. In *Proceedings of DA2PL'16*, pages 81–95, 2016a. *Cité page 63.*

Nawal Benabbou, Patrice Perny, and Paolo Viappiani. Incremental preference elicitation in multi-attribute domains for choice and ranking with the Borda count. In *Proceedings of SUM'16*, pages 81–95, 2016b. *Cité page 169.*

Nawal Benabbou, Patrice Perny, and Paolo Viappiani. Incremental elicitation of Choquet capacities for multicriteria choice, ranking and sorting problems. *Artificial Intelligence*, 246 :152–180, 2017. *Cité page 63.*

R. Benayoun, J. De Montgolfier, J. Tergny, and O. Laritchev. Linear programming with multiple objective functions : Step method (stem). *Mathematical programming*, 1(1) :366–375, 1971. *Cité page 51.*

Nadja Betzler, Arkadii Slinko, and Johannes Uhlmann. On the computation of fully proportional representation. *Journal of Artificial Intelligence Research*, pages 475–519, 2013. *Cité page 194.*

Michel Beuthe and Giuseppe Scannella. Comparative analysis of UTA multicriteria methods. *European Journal of operational research*, 130(2) :246–262, 2001. *Cité page 52.*

Duncan Black, Robert Albert Newing, Iain McLean, Alistair McMillan, and Burt L Monroe. *The theory of committees and elections*. Springer, 1958. *Cité page 33.*

Charles Blackorby, David Donaldson, and John A. Weymark. Social aggregation and the expected utility hypothesis. In Marc Fleurbaey, Maurice Salles, and John A. Weymark, editors, *Justice, political liberalism, and utilitarianism : themes from Harsanyi and Rawls*, pages 136–183. Cambridge University Press, 2008. *Cité page 218.*

Géraldine Bous and Marc Pirlot. Learning multicriteria utility functions with random utility models. In *Proceedings of ADT'13*, pages 101–115. Springer, 2013. *Cité page 52.*

Craig Boutilier. A POMDP Formulation of Preference Elicitation Problems. In *Proceedings of AAAI'02*, pages 239–246, Edmonton, 2002. *Cité pages 53, 54, 58, et 230.*

Craig Boutilier. Computational decision support : Regret-based models for optimization and preference elicitation. *CrowleyP. H. ZentallT. R.(Eds.), Comparative Decision Making : Analysis and Support Across Disciplines and Applications*, pages 423–453, 2013. *Cité page 12.*

Craig Boutilier, Thomas Dean, and Steve Hanks. Decision-theoretic planning : Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11(1) :94, 1999. *Cité page 214.*

Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, and David Poole. Preference-based constrained optimization with cp-nets. *Computational Intelligence*, 20(2) :137–157, 2004. *Cité page 122.*

Craig Boutilier, Relu Patrascu, Pascal Poupart, and Dale Schuurmans. Constraint-based Optimization and Utility Elicitation using the Minimax Decision Criterion. *Artifical Intelligence*, 170(8–9) :686–713, 2006. *Cité pages 13, 54, 55, 59, 89, 118, 122, et 230.*

Craig Boutilier, Ioannis Caragiannis, Simi Haber, Tyler Lu, Ariel D Procaccia, and Or Sheffet. Optimal social choice functions : A utilitarian view. *Artificial Intelligence*, 227 :190–213, 2015. *Cité page 193.*

Denis Bouyssou and Thierry Marchant. An axiomatic approach to noncompensatory sorting methods in MCDM, I : The case of two categories. *European Journal of Operational Research*, 178(1) :217 – 245, 2007a. *Cité page 105.*

Denis Bouyssou and Thierry Marchant. An axiomatic approach to noncompensatory sorting methods in MCDM, II : More than two categories. *European Journal of Operational Research*, 178(1) :246 – 276, 2007b. *Cité page 105.*

Ronen I. Brafman and Carmel Domshlak. Preference handling - an introductory tutorial. *AI Magazine*, 30(1) :58–86, 2009. *Cité page 122.*

Steven Brams and Peter C. Fishburn. *Approval voting.* Boston :Birkhäuser, 1983. *Cité pages 39 et 41.*

Steven J Brams. *Approval voting.* New York Times, 2016. *Cité page 40.*

Steven J Brams and Peter C. Fishburn. Approval voting. *American Political Science Review*, 72(03) : 831–847, 1978. *Cité page 39.*

Felix Brandt, Vincent Conitzer, and Ulle Endriss. Computational social choice. In *Chapter in G. Weiss (Ed.), Multiagent Systems*, pages 213–283. MIT Press, 2013. *Cité page 248.*

Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. Cambridge University Press, 2016. In press. *Cité pages 191 et 249.*

Jean-Pierre Brans and Philippe Vincke. A preference ranking organisation method : (the PROMETHEE method for multiple criteria decision-making). *Management science*, 31(6) :647–656, 1985. *Cité page 91.*

Darius Braziunas. *Decision-theoretic elicitation of generalized additive utilities.* PhD thesis, University of Toronto, 2011. *Cité pages 54, 57, et 64.*

Darius Braziunas and Craig Boutilier. Preference elicitation and generalized additive utility (nectar paper). In *Proceedings of AAAI'06*, Boston, MA, 2006. *Cité page 53.*

Darius Braziunas and Craig Boutilier. Minimax regret based elicitation of generalized additive utilities. In *Proceedings of UAI'07*, pages 25–32, 2007. *Cité pages 13, 54, 122, et 230.*

Darius Braziunas and Craig Boutilier. Assessing regret-based preference elicitation with the utpref recommendation system. In *Proceedings 11th ACM Conference on Electronic Commerce (EC-2010)*, pages 219–228, 2010. *Cité pages 12, 54, et 230.*

Colin F Camerer, George Loewenstein, and Matthew Rabin. *Advances in behavioral economics.* Princeton University Press, 2003. *Cité page 51.*

Ioannis Caragiannis and Ariel D Procaccia. Voting almost maximizes social welfare despite limited communication. *Artificial Intelligence*, 175(9) :1655–1671, 2011. *Cité page 193.*

Arthur Cayley. A theorem on trees. *The Quarterly Journal of Mathematics*, 23 :376–378, 1889. *Cité page 154.*

Davide P. Cervone, William V. Gehrlein, and William S. Zwicker. Which scoring rule maximizes Condorcet efficiency under IAC ? *Theory and Decision*, 58(2) :145–185, 2005. *Cité page 32.*

Urszula Chajewska and Daphne Koller. Utilities as random variables : Density estimation and structure discovery. In *Proceedings of UAI'00*, pages 63–71. Morgan Kaufmann Publishers Inc., 2000. *Cité page 230.*

Urszula Chajewska, Daphne Koller, and Ronald Parr. Making Rational Decisions Using Adaptive Utility Elicitation. In *Proceedings of AAAI'00*, pages 363–369, 2000. *Cité page 53.*

John R Chamberlin and Paul N Courant. Representative deliberations and representative decisions : Proportional representation and the Borda rule. *American Political Science Review*, 77(03) :718–733, 1983. *Cité pages 193 et 194.*

Alain Chateauneuf and Jean-Yves Jaffray. Some characterizations of lower probabilities and other monotone capacities through the use of Möbius inversion. *Mathematical Social Sciences*, 17(3) :263–283, 1989. *Cité pages 28 et 68.*

Alain Chateauneuf, Rose Anne Dana, and Jean-Marc Tallon. Diversification, convex preferences and non-empty core in the Choquet expected utility model. *Economic Theory*, 19(3) :509–523, 1999. *Cité page 27.*

Gustave Choquet. Theory of capacities. *Annales de l'Institut Fourier*, 5 :31–295, 1953. *Cité page 26.*

Edward H Clarke. Multipart pricing of public goods. *Public choice*, 11(1) :17–33, 1971. *Cité page 249.*

João CN Clímaco, José MF Craveirinha, and Marta MB Pascoal. An automated reference point-like approach for multicriteria shortest path problems. *Journal of Systems Science and Systems Engineering*, 15(3) :314–329, 2006. *Cité page 12.*

Javier Coego, Lawrence Mandow, and JL Pérez De La Cruz. A new approach to iterative deepening multiobjective A*. In *Congress of the Italian Association for Artificial Intelligence*, pages 264–273. Springer, 2009. *Cité page 128.*

Javier Coego, Lawrence Mandow, and JL Pérez De La Cruz. A comparison of multiobjective depth-first algorithms. *Journal of Intelligent Manufacturing*, 24(4) :821–829, 2013. *Cité page 128.*

Michele Cohen, Jean-Yves Jaffray, and Tanios Said. Experimental comparison of individual behavior under risk and under uncertainty for gains and for losses. *Organizational behavior and human decision processes*, 39(1) :1–22, 1987. *Cité page 45.*

William W. Cohen, Robert E. Schapire, and Yoram Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10(1) :243–270, 1999. *Cité page 91.*

Elías F Combarro and Pedro Miranda. On the polytope of non-additive measures. *Fuzzy Sets and Systems*, 159(16) :2145–2162, 2008. *Cité page 68.*

Marie Jean Antoine de Caritat Marquis de Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions : rendues à la pluralité des voix*. De l'imprimerie royale, 1785. *Cité page 32.*

Vincent Conitzer and Tuomas Sandholm. Vote elicitation : complexity and strategy-proofness. In *Proceedings of AAAI'02*, pages 392–397, 2002. *Cité page 247.*

Vincent Conitzer and Tuomas Sandholm. Communication complexity of common voting rules. In *Proceedings 6th ACM Conference on Electronic Commerce (EC-2005)*, pages 78–87, 2005. *Cité page 247.*

H. W. Corley. Efficient spanning trees. *Journal of optimization theory and applications*, 45(3) :481–485, 1985. *Cité page 155.*

Douglas E. Critchlow, Michael A. Fligner, and Joseph S. Verducci. Probability models on rankings. *Journal of Mathematical Psychology*, 35(3) :294 – 318, 1991. *Cité page 91.*

Pallab Dasgupta, P.P. Chakrabarti, and S.C. De Sarkar. Utility of *pathmax* in partial order heuristic search. *J. of algorithms*, 55 :317–322, 1995. *Cité pages 123 et 128.*

Bruno De Finetti. La prévision : ses lois logiques, ses sources subjectives. In *Annales de l'institut Henri Poincaré*, volume 7, pages 1–68, 1937. *Cité page 41.*

Rina Dechter. From local to global consistency. *Artificial intelligence*, 55 :87–107, 1992. *Cité page 74.*

Arthur P. Dempster. Upper and lower probabilities induced by a multivalued mapping. *The annals of mathematical statistics*, pages 325–339, 1967. *Cité pages 41 et 68.*

Lihi Naamani Dery, Meir Kalech, Lior Rokach, and Bracha Shapira. Reaching a joint decision with minimal elicitation of voter preferences. *Information Sciences*, 278 :466–487, 2014. *Cité pages 54, 170, et 193.*

Lihi Naamani Dery, Meir Kalech, Lior Rokach, and Bracha Shapira. Reducing preference elicitation in group decision making. *Expert Syst. Appl.*, 61 :246–261, 2016. *Cité pages 54, 170, et 193.*

JM Devaud, G Groussaud, and Eric Jacquet-Lagreze. UTADIS : Une méthode de construction de fonctions d'utilité additives rendant compte de jugements globaux. *European Working Group on Multicriteria Decision Aid, Bochum*, 1980. *Cité page 94.*

Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1) : 269–271, 1959. *Cité page 126.*

Ning Ding and Fangzhen Lin. Voting with partial information : what questions to ask ? In *Proceedings of AAMAS'13*, pages 1237–1238, 2013. *Cité pages 170 et 193.*

Carmel Domshlak, Eyke Hüllermeier, Souhila Kaci, and Henri Prade. Preferences in AI : an overview. *Artificial Intelligence Journal*, 175(7-8) :1037–1052, 2011. *Cité page 122.*

Joanna Drummond and Craig Boutilier. Elicitation and approximately stable matching with partial preferences. In *Proceedings of IJCAI'13*, pages 97–105, 2013. *Cité page 59.*

Joanna Drummond and Craig Boutilier. Preference elicitation and interview minimization in stable matchings. In *Proceedings of AAAI'14*, pages 645–653, 2014. *Cité pages 13 et 122.*

John S Dryzek and Christian List. Social choice theory and deliberative democracy : a response to aldred. *British Journal of Political Science*, 34(04) :752–758, 2004. *Cité page 33.*

Didier Dubois and Henri Prade. *Possibility theory : an approach to computerized processing of uncertainty.* New York : Plenum Press, 1988. *Cité page 41.*

Matthias Ehrgott. *Multicriteria optimization.* Springer Science & Business Media, 2006. *Cité pages 17, 19, 155, et 165.*

Matthias Ehrgott and Xavier Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR-Spektrum*, 22(4) :425–460, 2000. *Cité page 154.*

Edith Elkind and Anisse Ismaili. OWA-based extensions of the chamberlin courant rule. In *Proceedings of ADT'15*, pages 486–502. Springer, 2015. *Cité page 26.*

Edith Elkind, Piotr Faliszewski, Piotr Skowron, and Arkadii Slinko. Properties of multiwinner voting rules. In *Proceedings of AAMAS'14*, pages 53–60, 2014. *Cité pages 191 et 194.*

Ulle Endriss. Vote manipulation in the presence of multiple sincere ballots. In *Proceedings of the 11th conference on Theoretical aspects of rationality and knowledge*, pages 125–134. ACM, 2007. *Cité page 40.*

Funda Ergun, Rakesh Sinha, and Lisa Zhang. An improved fptas for restricted shortest path. *Information Processing Letters*, 83(5) :287–291, 2002. *Cité page 142.*

Bruno Escoffier, Jérôme Lang, and Meltem Öztürk. Single-peaked consistency and its complexity. In *Proceedings of ECAI'08*, volume 8, pages 366–370, 2008. *Cité page 33.*

Hélène Fargier and Jérôme Lang. Uncertainty in constraint satisfaction problems : a probabilistic approach. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 97–104. Springer, 1993. *Cité page 248.*

Peter C. Fishburn. Methods of estimating additive utilities. *Management science*, 13(7) :435–453, 1967. *Cité page 50.*

Peter C. Fishburn. Utility theory. *Management science*, 14(5) :335–378, 1968. *Cité pages 55 et 140.*

Peter C. Fishburn. *Utility theory for decision making.* Publications in operations research. Wiley, 1970. *Cité page 91.*

Peter C. Fishburn. Axioms for approval voting : direct proof. *Journal of Economic Theory*, 19(1) : 180–185, 1978. *Cité page 39.*

Peter C. Fishburn and Steven J. Brams. Approval voting, condorcet's principle, and runoff elections. *Public Choice*, 36(1) :89–114, 1981. *Cité page 41.*

Peter C. Fishburn and William V. Gehrlein. Borda's rule, positional voting, and condorcet's simple majority principle. *Public Choice*, 28(1) :79–88, 1976. *Cité page 37.*

Peter C. Fishburn and Gary A. Kochenberger. Two-piece von neumann-morgenstern utility functions. *Decision Sciences*, 10(4) :503–518, 1979. *Cité page 238.*

Marc Fleurbaey and Philippe Mongin. The utilitarian relevance of the aggregation theorem. *American Economic Journal : Microeconomics*, 8(3) :289–306, 2016. *Cité page 218.*

Simon French. *Decision theory : an introduction to the mathematics of rationality.* Halsted Press, 1986. *Cité page 57.*

Johannes Fürnkranz and Eyke Hüllermeier. Preference learning : An introduction. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 1–17. Springer Berlin Heidelberg, 2011. *Cité page 94.*

Johannes Fürnkranz, Eyke Hüllermeier, and Stijn Vanderlooy. Binary decomposition methods for multipartite ranking. In W. Buntine, M. Grobelnik, D. Mladenić, and J. Shawe-Taylor, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5781 of *Lecture Notes in Computer Science*, pages 359–374. Springer Berlin Heidelberg, 2009. *Cité page 94.*

Lucie Galand and Patrice Perny. Search for Compromise Solutions in Multiobjective State Space Graphs. In *Proceedings of ECAI'06*, pages 93–97, 2006. *Cité pages 12 et 123.*

Lucie Galand and Patrice Perny. Search for Choquet-optimal paths under uncertainty. In *Proceedings of UAI'07*, pages 125–132, 2007. *Cité pages 12 et 123.*

Lucie Galand and Olivier Spanjaard. Exact algorithms for OWA-optimization in multiobjective spanning tree problems. *Computers and Operations Research*, 39(7) :1540–1554, 2012. *Cité page 12.*

Lucie Galand, Julien Lesca, and Patrice Perny. Dominance Rules for the Choquet Integral in Multiobjective Dynamic Programming. In *Proceedings of IJCAI'13*, pages 538–544, 2013. *Cité pages 12, 122, 123, et 142.*

William V Gehrlein. Condorcet's paradox. *Theory and Decision*, 15(2) :161–197, 1983. *Cité page 32.*

William V Gehrlein, Dominique Lepelley, and Hatem Smaoui. The condorcet efficiency of voting rules with mutually coherent voter preferences : a borda compromise. *Annals of Economics and Statistics/Annales d'Économie et de Statistique*, pages 107–125, 2011. *Cité page 32.*

Mirco Gelain, Maria Silvia Pini, Francesca Rossi, K. Brent Venable, and Toby Walsh. Elicitation strategies for soft constraint problems with missing preferences : Properties, algorithms and experimental studies. *Artificial Intelligence Journal*, 174(3-4) :270–294, 2010. *Cité page 122.*

Malik Ghallab and Dennis G Allard. A_ε- an efficient near admissible heuristic search algorithm. In *Proceedings of IJCAI'83*, pages 789–791, 1983. *Cité pages 141 et 142.*

Soumyadip Ghosh and Jayant Kalagnanam. Polyhedral sampling for multiattribute preference elicitation. In *Proceedings of the 4th ACM conference on Electronic Commerce*, pages 256–257. ACM, 2003. *Cité page 55.*

Allan Gibbard. Manipulation of voting schemes : a general result. *Econometrica : journal of the Econometric Society*, pages 587–601, 1973. *Cité page 35.*

Hugo Gilbert, Olivier Spanjaard, Paolo Viappiani, and Paul Weng. Reducing the number of queries in interactive value iteration. In *Proceedings of ADT'15*, pages 139–152, 2015. *Cité pages 13 et 122.*

Claudia V Goldman and Shlomo Zilberstein. Communication-based decomposition mechanisms for decentralized MDPs. *Journal of Artificial Intelligence Research (JAIR)*, 32 :169–202, 2008. *Cité page 214.*

Judy Goldsmith, Jérôme Lang, Nicholas Mattei, and Patrice Perny. Voting with rank dependent scoring rules. Citeseer, 2014. *Cité page 26.*

Christophe Gonzales and Patrice Perny. GAI Networks for Utility Elicitation. In *Proceedings of the 9th International Conference on the Principles of Knowledge Representation and Reasoning*, pages 224–234, 2004. *Cité pages 13 et 122.*

Robert E Goodin and Christian List. A conditional defense of plurality rule : generalizing may's theorem in a restricted informational environment. *American Journal of Political Science*, 50(4) :940–949, 2006. *Cité page 39.*

Michel Grabisch. The application of fuzzy integrals in multicriteria decision making. *European journal of operational research*, 89(3) :445–456, 1996. *Cité page 29.*

Michel Grabisch and Christophe Labreuche. A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1) :247–286, 2010. *Cité pages 12 et 26.*

Michel Grabisch and Pedro Miranda. On the vertices of the k-additive core. *Discrete Mathematics*, 308 (22) :5204–5217, 2008. *Cité page 68.*

Michel Grabisch and Patrice Perny. Agrégation multicritère. *Logique floue, principes, aide à la décision*, pages 80–120, 2003. *Cité pages 20, 94, et 105.*

Michel Grabisch, Hunt T. Nguyen, and Elbert A. Walker. *Fundamentals of Uncertainty Calculi, with Applications.* Encyclopedia of Mathematics and its Applications. chap. Identification and Interpretation of Fuzzy Measures . Kluwer Academic Publishers, 1995. *Cité pages 52 et 72.*

Michel Grabisch, Ivan Kojadinovic, and Patrick Meyer. A review of methods for capacity identification in Choquet integral based multi-attribute utility theory. *European Journal of Operational Research*, 186(2) :766–785, 2008. *Cité page 52.*

Michel Grabisch, Jean-Luc Marichal, Radko Mesiar, and Endre Pap. *Aggregation Functions.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, New-York, 2009. *Cité pages 29 et 72.*

Salvatore Greco, Vincent Mousseau, and Roman Słowiński. Ordinal regression revisited : multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research*, 191(2) : 416–436, 2008. *Cité page 91.*

Salvatore Greco, Matthias Ehrgott, and José Rui Figueira. *Trends in multiple criteria decision analysis.* Chapitre 8, Springer Science & Business Media, 2010a. *Cité page 16.*

Salvatore Greco, Vincent Mousseau, and Roman Slowinski. Multiple criteria sorting with a set of additive value functions. *European Journal of Operational Research*, 207(3) :1455–1470, 2010b. *Cité page 94.*

Salvatore Greco, Roman Słowiński, José Rui Figueira, and Vincent Mousseau. Robust ordinal regression. In *Trends in multiple criteria decision analysis*, pages 241–283. Springer, 2010c. *Cité page 53.*

Salvatore Greco, Vincent Mousseau, and Roman Słowiński. Robust ordinal regression for value functions handling interacting criteria. *European Journal of Operational Research*, 239(3) :711–730, 2014. *Cité page 53.*

Salvatore Greco, Matthias Ehrgott, and José Rui Figueira. *Multiple criteria decision analysis : state of the art surveys.* Springer International Series in Operations Research & Management Science, 2016. *Cité pages 17, 19, et 51.*

Theodore Groves. Incentives in teams. *Econometrica : Journal of the Econometric Society*, pages 617–631, 1973. *Cité page 249.*

Carlos Guestrin, Daphne Koller, and Ronald Parr. Multiagent planning with factored MDPs. In *Proceedings of NIPS'01*, volume 1, pages 1523–1530, 2001. *Cité page 214.*

Horst W. Hamacher and Günter Ruhe. On spanning tree problems with multiple objectives. *Annals of Operations Research*, 52(4) :209–230, 1994. *Cité pages 20, 154, 155, et 159.*

Peter J. Hammond. Consequentialist foundations for expected utility. *Theory and decision*, 25(1) :25–78, 1988. *Cité page 219.*

Peter J. Hammond. Harsanyi's utilitarian theorem : A simpler proof and some ethical connotations. In *Rational Interaction*, pages 305–319. Springer, 1992. *Cité page 218.*

Pierre Hansen. Bicriterion path problems. In *Multiple criteria decision making theory and application*, pages 109–127. Springer, 1980. *Cité pages 20, 136, 142, et 143.*

S. Harikumar and Shashi Kumar. Iterative deepening multiobjective A*. *Information Processing Letters*, 58 :11–15, 1996. *Cité page 128.*

John C Harsanyi. Bayesian decision theory and utilitarian ethics. *The American Economic Review*, 68 (2) :223–228, 1978. *Cité page 218.*

Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Man, and Cybernetics*, 4(2) :100–107, 1968. *Cité pages 123 et 124.*

R. Herbrich, T. Graepel, and K. Obermayer. *Large Margin Rank Boundaries for Ordinal Regression*, chapter 7, pages 115–132. MIT Press, 2000. *Cité page 91.*

Ronald A Howard and James E Matheson. *Readings on the Princinples and Applications of Decision Analysis*. Volume II. Strategic Decision Group, Menlo Park, CA, 1984. *Cité page 214.*

Fabien Le Huédé, Michel Grabisch, Christophe Labreuche, and Pierre Savéant. Integration and propagation of a multi-criteria decision making model in constraint programming. *Journal of Heuristics*, 12 (4-5) :329–346, 2006. *Cité pages 52 et 72.*

Eyke Hüllermeier and Ali Fallah Tehrani. Efficient learning of classifiers based on the 2-additive Choquet integral. In *Computational Intelligence in Intelligent Data Analysis Studies in Computational Intelligence Volume*, volume 445, pages 17–29, 2013. *Cité pages 69 et 72.*

Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, and Klaus Brinker. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172(16–17) :1897 – 1916, 2008. *Cité page 91.*

Leonid Hurwicz. Optimality criteria for decision making under ignorance. *Cowles Commission Discussion Paper, Statistics, tm. 370*, 1951. *Cité page 57.*

Oscar H Ibarra and Chul E Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM (JACM)*, 22(4) :463–468, 1975. *Cité page 203.*

Jonathan E Ingersoll. *Theory of financial decision making*, volume 3. Rowman & Littlefield, 1987. *Cité page 44.*

Vijay S Iyengar, Jon Lee, and Murray Campbell. Evaluating multiple attribute items using queries. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 144–153. ACM, 2001. *Cité page 55.*

Eric Jacquet-Lagrèze. An application of the UTA discriminant model for the evaluation of R&D projects. In *Advances in Multicriteria Analysis*, Nonconvex Optimization and Its Applications, pages 203–211. Springer US, 1995. *Cité page 94.*

Eric Jacquet-Lagrèze and Jean Siskos. Assessing a set of additive utility functions for multicriteria decision-making, the UTA method. *European journal of operational research*, 10(2) :151–164, 1982. *Cité pages 52 et 91.*

Gildas Jeantet. *Algorithmes pour la décision séquentielle dans l'incertain : optimisation de l'utilité espérée dépendant du rang et du critère de Hurwicz.* PhD thesis, Université Pierre et Marie Curie, 2010. *Cité page 227.*

Gildas Jeantet, Patrice Perny, and Olivier Spanjaard. Sequential Decision Making with Rank Dependent Utility : a Minimax Regret Approach. In *Proceedings of AAAI'12*, pages 1931–1937, 2012. *Cité pages 12 et 248.*

Sami Kaddani, Daniel Vanderpooten, Jean-Michel Vanpeperstraete, and Hassene Aissi. Weighted sum model with partial preference information : application to multi-objective optimization. *European Journal of Operational Research*, 2017. *Cité pages 13 et 122.*

Daniel Kahneman and Amos Tversky. Prospect theory : An analysis of decision under risk. *Econometrica : Journal of the econometric society*, pages 263–291, 1979. *Cité pages 49 et 238.*

Meir Kalech, Sarit Kraus, and Gal A. Kaminka. Practical voting rules with partial information. *Autonomous Agents and Multi-Agent Systems*, 22(1) :151–182, 2010. *Cité pages 54, 170, et 193.*

Ralph L. Keeney and Howard Raiffa. *Decisions with multiple objectives : Preferences and value tradeoffs.* J. Wiley, New York, 1976. *Cité pages 50, 55, 91, 141, et 230.*

John Kellett and Kenneth Mott. Presidential primaries : Measuring popular choice. *Polity*, pages 528–537, 1977. *Cité page 39.*

Daniel Kikuti, Fabio Gagliardi Cozman, and Ricardo Shirota Filho. Sequential decision making with partially ordered preferences. *Artificial Intelligence*, 175(7) :1346–1365, 2011. *Cité page 220.*

D. Marc Kilgour. Approval balloting for multi-winner elections. In *Handbook on approval voting*, pages 105–124. Springer, 2010. *Cité page 193.*

Ronald Klimberg. Grads : A new graphical display system for visualizing multiple criteria solutions. *Computers & operations research*, 19(7) :707–711, 1992. *Cité page 52.*

Frank Knight. *Risk, Uncertainty and Profit.* Houghton Mifflin, 1921. *Cité page 41.*

Joshua D Knowles and David W Corne. Enumeration of pareto optimal multi-criteria spanning trees–a proof of the incorrectness of zhou and gen's proposed algorithm. *European journal of operational research*, 143(3) :543–547, 2002. *Cité page 155.*

Veronika Köbberling and Peter P. Wakker. Preference foundations for nonexpected utility : A generalized and simplified technique. *Mathematics of Operations Research*, 28(3) :395–423, 2003. *Cité page 27.*

Ivan Kojadinovic. Minimum variance capacity identification. *European Journal of Operational Research*, 177(1) :498–514, 2007. *Cité page 52.*

Kathrin Konczak and Jérôme Lang. Voting procedures with incomplete preferences. In *Proceedings of IJCAI'05 Multidisciplinary Workshop on Advances in Preference Handling*, volume 20, 2005. *Cité pages 122, 170, et 193.*

Pekka Korhonen. *Multiple criteria decision analysis.* chap. Interactive Methods. Greco, Salvatore and Figueira, J and Ehrgott, M. Springer, 2005. *Cité pages 13, 51, et 122.*

Pekka Korhonen, Herbert Moskowitz, and Jyrki Wallenius. Choice behavior in interactive multiple-criteria decision making. *Annals of Operations Research*, 23(1) :161–179, 1990. *Cité page 26.*

Pekka J Korhonen and Jukka Laakso. A visual interactive method for solving the multiple criteria problem. *European Journal of Operational Research*, 24(2) :277–287, 1986. *Cité page 51.*

Frédéric Koriche and Bruno Zanuttini. Learning conditional preference networks. *Artif. Intell.*, 174(11) : 685–703, 2010. *Cité pages 13 et 122.*

Panos Kouvelis and Gang Yu. *Robust Discrete Optimization and Its Applications.* Kluwer, Dordrecht, 1997. *Cité page 55.*

David H. Krantz, R. Duncan Luce, Patrick Suppes, and Amos Tversky. *Foundations of measurement.* Academic Press, New York, 1971. *Cité page 50.*

Joseph B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7 :48–50, 1956. *Cité pages 153 et 155.*

Eyal Kushilevitz and Noam Nisan. *Communication Complexity.* Cambridge University Press, 1997. *Cité page 247.*

Christophe Labreuche. An axiomatization of the choquet integral and its utility functions without any commensurability assumption. In *Advances in Computational Intelligence*, pages 258–267. Springer, 2012. *Cité page 27.*

Jérôme Lang and Lirong Xia. Sequential composition of voting rules in multi-issue domains. *Mathematical social sciences*, 57(3) :304–324, 2009. *Cité page 170.*

Jérôme Lang, Maria Silvia Pini, Francesca Rossi, Domenico Salvagnin, Kristen Brent Venable, and Toby Walsh. Winner determination in voting trees with incomplete preferences and weighted votes. *Autonomous Agents and Multi-Agent Systems*, 25(1) :130–157, 2012. *Cité pages 170 et 193.*

Eugene L Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4(4) :339–356, 1979. *Cité page 203.*

A Lewandowski and J Granat. Dynamic biplot as the interaction interface for aspiration based decision support systems. *Lecture Notes in Economics and Mathematical Systems*, 356 :229–241, 1991. *Cité page 51.*

Christian List, Robert C Luskin, James S Fishkin, and Iain McLean. Deliberation, single-peakedness, and the possibility of meaningful democracy : evidence from deliberative polls. *the Journal of Politics*, 75(01) :80–95, 2013. *Cité page 33.*

Bonifacio Llamazares. On generalizations of weighted means and OWA operators. In *EUSFLAT Conf.*, pages 9–14, 2011. *Cité pages 26 et 248.*

László Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pages 235–257. Springer, 1983. *Cité pages 27 et 28.*

Tyler Lu and Craig Boutilier. Vote elicitation with probabilistic preference models : Empirical estimation and cost tradeoffs. In *Proceedings of ADT'11*, pages 135–149, 2011a. *Cité page 170.*

Tyler Lu and Craig Boutilier. Robust approximation and incremental elicitation in voting protocols. In *Proceedings of IJCAI'11*, pages 287–293, 2011b. *Cité pages 54, 59, 170, 172, 174, 185, 193, et 210.*

Tyler Lu and Craig Boutilier. Budgeted social choice : From consensus to personalized decision making. In *Proceedings of IJCAI'11*, volume 11, pages 280–286, 2011c. *Cité pages 191 et 193.*

Tyler Lu and Craig Boutilier. Multi-winner social choice with incomplete preferences. In *Proceedings of IJCAI'13*, pages 263–270, 2013. *Cité page 194.*

Mark J Machina. " expected utility" analysis without the independence axiom. *Econometrica : Journal of the Econometric Society*, pages 277–323, 1982. *Cité page 49.*

Lawrence Mandow and José-Luis Pérez de la Cruz. Comparison of heuristics in multiobjective a* search. In *Conference of the Spanish Association for Artificial Intelligence*, pages 180–189. Springer, 2005a. *Cité page 128.*

Lawrence Mandow and José-Luis Pérez de la Cruz. A new approach to multiobjective A* search. In *Proceedings of IJCAI'05*, pages 218–223, 2005b. *Cité pages 122, 124, 126, 128, 132, et 167.*

Jean-Luc Marichal. An axiomatic approach of the discrete choquet integral as a tool to aggregate interacting criteria. *IEEE transactions on fuzzy systems*, 8(6) :800–807, 2000. *Cité page 27.*

Jean-Luc Marichal and Marc Roubens. Determination of weights of interacting criteria from a reference set. *European Journal of Operational Research*, 124(3) :641–650, 2000. *Cité pages 52 et 72.*

Jean-Luc Marichal, Patrick Meyer, and Marc Roubens. Sorting multi-attribute alternatives : the TOMASO method. *Computers & Operations Research*, 32(2) :861–877, 2005. *Cité pages 52 et 72.*

Radu Marinescu, Abdul Razak, and Nic Wilson. Multi-objective constraint optimization with tradeoffs. In *International Conference on Principles and Practice of Constraint Programming*, pages 497–512. Springer, 2013. *Cité page 122.*

Kenneth May. A set of independent necessary and sufficient conditions for simple majority decision. *Econometrica : Journal of the Econometric Society*, pages 680–684, 1952. *Cité page 31.*

Reshef Meir, Ariel D Procaccia, Jeffrey S Rosenschein, and Aviv Zohar. Complexity of strategic behavior in multi-winner elections. *Journal of Artificial Intelligence Research (JAIR)*, 33 :149–178, 2008. *Cité page 193.*

Patrick Meyer and Marc Roubens. Choice, ranking and sorting in fuzzy multiple criteria decision aid. In *Multiple criteria decision analysis : State of the art surveys*, pages 471–503. Springer, 2005. *Cité page 52.*

Patrick Meyer and Marc Roubens. On the use of the Choquet integral with fuzzy numbers in multiple criteria decision support. *Fuzzy Sets and Systems*, 157(7) :927–938, 2006. *Cité pages 52, 68, et 72.*

Kaisa Miettinen. *Nonlinear multiobjective optimization*, volume 12. International Series in Operations Research and Management Science, Kluwer Academic Publishers, Dordrecht, 1999. *Cité page 51.*

David Miller. Deliberative democracy and social choice. *Political studies*, 40(s1) :54–67, 1992. *Cité page 33.*

John Mingers and Jonathan Rosenhead. Problem structuring methods in action. *European Journal of Operational Research*, 152(3) :530–554, 2004. *Cité page 16.*

Philippe Mongin. Harsanyi's aggregation theorem : multi-profile version and unsettled questions. *Social Choice and Welfare*, 11(4) :331–354, 1994. *Cité page 218.*

Burt L Monroe. Fully proportional representation. *American Political Science Review*, 89(04) :925–940, 1995. *Cité page 193.*

Richard A Morin. *Structural Reform : Ballots*. Vantage Press, 1980. *Cité page 39.*

Hervé Moulin. On strategy-proofness and single peakedness. *Public Choice*, 35(4) :437–455, 1980. *Cité page 36.*

Hervi Moulin. *Axioms of cooperative decision making.* Number 15. Cambridge University Press, 1991. *Cité page 37.*

Vincent Mousseau and Roman Slowinski. Inferring an ELECTRE-TRI model from assignment examples. *Journal of Global Optimization*, 12(2) :157–174, 1998. *Cité pages 105 et 108.*

Vincent Mousseau, Roman Slowinski, and Piotr Zielniewicz. A useroriented implementation of the ELECTRE-TRI method integrating preference elicitation support. *Computers and Operations Research*, 27 :757–777, 2000. *Cité page 108.*

Toshiaki Murofushi and Tatsuya Mori. An analysis of evaluation model using fuzzy measure and the Choquet integral. In *Proceedings of 5th Fuzzy System Symposium*, pages 207–212, 1989. *Cité page 52.*

Thomas D. Nielsen and Jean-Yves Jaffray. Dynamic decision making without expected utility : An operational approach. *European Journal of Operational Research*, 169(1) :226–246, 2006. *Cité page 12.*

Richard G Niemi. The problem of strategic behavior under approval voting. *American Political Science Review*, 78(04) :952–958, 1984. *Cité page 41.*

Noam Nisan and Ilya Segal. The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory*, 129(1) :192–224, 2006. *Cité page 248.*

Wlodzimierz Ogryczak. Inequality measures and equitable approaches to location problems. *European Journal of Operational Research*, 122(2) :374–391, 2000. *Cité page 26.*

Włodzimierz Ogryczak and Tomasz Śliwiński. On solving linear programs with the ordered weighted averaging objective. *European Journal of Operational Research*, 148(1) :80–91, 2003. *Cité page 12.*

Włodzimierz Ogryczak and Tomasz Śliwiński. On efficient WOWA optimization for decision support under risk. *International Journal of Approximate Reasoning*, 50(6) :915–928, 2009. *Cité page 12.*

Joel Oren and Brendan Lucier. Online (budgeted) social choice. *Proceedings of AAAI'14*, pages 1456–1462, 2014. *Cité page 191.*

Guy Ottowell. The arithmetic of voting. *Defense of Variety*, 4 :42–44, 1977. *Cité page 39.*

Christos H Papadimitriou and Mihalis Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 86–92. IEEE, 2000. *Cité page 142.*

Judea Pearl and Jin H Kim. Studies in semi-admissible heuristics. *IEEE transactions on pattern analysis and machine intelligence*, (4) :392–399, 1982. *Cité pages 141 et 142.*

Bart Peintner, Paolo Viappiani, and Neil Yorke-Smith. Preferences in interactive systems : Technical challenges and case studies. *AI Magazine*, 29(4) :13–24, 2008. *Cité page 12.*

Patrice Perny. Multicriteria filtering methods based on concordance and non-discordance principles. *Annals of operations Research*, 80 :137–165, 1998. *Cité pages 105, 106, et 107.*

Patrice Perny. *Modélisation des préférences, agrégation multicritère et systèmes d'aide à la décision.* 2000. *Cité pages 20, 94, et 105.*

Patrice Perny and Olivier Spanjaard. A Preference-Based Approach to Spanning Trees and Shortest Paths Problems. *European Journal of Operational Research*, 162(3) :584–601, 2005. *Cité pages 122, 155, et 221.*

Patrice Perny and Olivier Spanjaard. Near Admissible Algorithms for Multiobjective Search. In *Proceedings of ECAI'08*, pages 490–494, 2008. *Cité pages 142 et 143.*

Patrice Perny, Paolo Viappiani, and Abdellah Boukhatem. Incremental Preference Elicitation for Decision Making Under Risk with the Rank-Dependent Utility Model. In *Proceedings of UAI'16*, 2016. *Cité pages 238 et 249.*

Richard F Potthoff and Steven J Brams. Proportional representation broadening the options. *Journal of Theoretical Politics*, 10(2) :147–178, 1998. *Cité page 193.*

R. C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36 : 1389–1401, 1957. *Cité pages 153, 154, 156, et 167.*

Ariel D Procaccia and Jeffrey S Rosenschein. The communication complexity of coalition formation among autonomous agents. In *Proceedings of AAMAS'06*, pages 505–512. ACM, 2006a. *Cité page 248.*

Ariel D. Procaccia and Jeffrey S. Rosenschein. The distortion of cardinal preferences in voting. In *Proceedings of the Tenth International Workshop on Cooperative Information Agent*, pages 317–331. Springer, 2006b. *Cité page 193.*

Ariel D Procaccia, Jeffrey S Rosenschein, and Aviv Zohar. On the complexity of achieving proportional representation. *Social Choice and Welfare*, 30(3) :353–362, 2008. *Cité page 193.*

Jose R. Quevedo, Elena Montañés, Oscar Luaces, and Juan J. del Coz. Adapting decision DAGs for multipartite ranking. In J. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 6323 of *Lecture Notes in Computer Science*, pages 115–130. Springer Berlin Heidelberg, 2010. *Cité page 94.*

John Quiggin. *Generalized expected utility theory : The rank dependent model.* Springer Science & Business Media, 1992. *Cité pages 12, 49, 238, et 248.*

Howard Raiffa. *Decision analysis : introductory lectures on choices under uncertainty.* Addison-Wesley, 1968. *Cité pages 46 et 214.*

James O. Ramsay. Monotone regression spline in action. *Statistical Science*, page 425–441, 1988. *Cité pages 234, 235, et 237.*

Kevin Regan and Craig Boutilier. Eliciting additive reward functions for markov decision processes. In *Proceedings of IJCAI'11*, pages 2159–2164, 2011. *Cité pages 13 et 122.*

Michel Regenwetter. *Behavioral social choice : probabilistic models, statistical inference, and applications.* Cambridge University Press, 2006. *Cité page 32.*

Diederik M. Roijers, Shimon Whiteson, and Frans A. Oliehoek. Linear support for multi-objective coordination graphs. In *Proceedings AAMAS'14*, pages 1297–1304, 2014. *Cité page 122.*

Jonathan Rosenhead and John Mingers. *Rational analysis for a problematic world revisited.* John Wiley and Sons, 2001. *Cité page 16.*

Francesca Rossi, Peter Van Beek, and Toby Walsh. *Handbook of constraint programming.* Elsevier, 2006. *Cité page 248.*

Francesca Rossi, Kristen Brent Venable, and Toby Walsh. Preferences in constraint satisfaction and optimization. *AI magazine*, 29(4) :58, 2009. *Cité page 248.*

Francesca Rossi, Kristen Brent Venable, and Toby Walsh. A short introduction to preferences : between artificial intelligence and social choice. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 5(4) :1–102, 2011. *Cité page 170.*

Gian-Carlo Rota. On the foundations of combinatorial theory i. theory of möbius functions. *Probability theory and related fields*, 2(4) :340–368, 1964. *Cité page 28.*

Michael Rothschild and Joseph E Stiglitz. Increasing risk : I. A definition. *Journal of Economic theory*, 2(3) :225–243, 1970. *Cité page 45.*

Bernard Roy. A multicriteria analysis for trichotomic segmentation problems. In *Multiple Criteria Analysis*, pages 245–257. P. Nijkamp and J. Spronk (eds), Gaver, 1981. *Cité page 105.*

Bernard Roy. *Multicriteria Methodology for Decision Analysis.* Kluwer Academic Publishers, 1996. *Cité pages 13, 16, et 91.*

Donald G. Saari. Condorcet domains : A geometric perspective. In *The Mathematics of Preference, Choice and Order*, pages 161–182. Springer, 2009. *Cité page 32.*

Thomas L. Saaty. *The Analytic Hierarchy Process, Planning, Piority Setting, Resource Allocation.* McGraw-Hill, New york, 1980. *Cité page 91.*

Ahti Salo and Raimo P. Hämäläinen. Preference ratios in multiattribute evaluation (PRIME)–elicitation and decision procedures under incomplete information. *IEEE Trans. on Systems, Man and Cybernetics*, 31(6) :533–545, 2001. *Cité page 55.*

Mark Allen Satterthwaite. Strategy-proofness and arrow's conditions : Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of economic theory*, 10(2) : 187–217, 1975. *Cité page 35.*

Leonard J. Savage. *The Foundations of Statistics*. Wiley, New York, 1954. *Cité pages 41 et 55.*

Thomas Schiex, Helene Fargier, Gerard Verfaillie, et al. Valued constraint satisfaction problems : Hard and easy problems. *IJCAI'95*, 95 :631–639, 1995. *Cité page 248.*

David Schmeidler. Integral representation without additivity. *Proceedings of the American Mathematical Society*, 97(2) :255–261, 1986. *Cité pages 12, 26, 27, 41, et 140.*

Florian Seipp. *On Adjacency, Cardinality, and Partial Dominance in Discrete Multiple Objective Optimization*. PhD thesis, Technischen Universitat Kaiserslautern, 2013. *Cité page 167.*

Amartya Sen. Social choice theory. *Handbook of mathematical economics*, 3 :1073–1181, 1986. *Cité page 218.*

Paolo Serafini. Some considerations about computational complexity for multi objective combinatorial problems. In *Recent advances and historical development of vector optimization*, pages 222–232. Springer, 1987. *Cité page 155.*

Murat R Sertel. Characterizing approval voting. *Journal of Economic Theory*, 45(1) :207–211, 1988. *Cité page 39.*

Glenn Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976. *Cité pages 28, 41, et 68.*

Anthony F. Shorrocks. Ranking income distributions. *Economica*, 50(197) :3–17, 1983. *Cité page 25.*

Herbert A Simon. A behavioral model of rational choice. *The quarterly journal of economics*, 69(1) : 99–118, 1955. *Cité page 51.*

Yannis Siskos and Denis Yannacopoulos. UTASTAR : An ordinal regression method for building additive value functions. *Investigação Operacional*, 5(1) :39–53, 1985. *Cité page 52.*

Piotr Skowron, Piotr Faliszewski, and Jérôme Lang. Finding a collective set of items : From proportional multirepresentation to group recommendation. In *Proceedings of AAAI'15*, pages 2131–2137, 2015a. *Cité page 191.*

Piotr Skowron, Lan Yu, Piotr Faliszewski, and Edith Elkind. The complexity of fully proportional representation for single-crossing electorates. *Theoretical Computer Science*, 569 :43–57, 2015b. *Cité page 194.*

John H Smith. Aggregation of preferences with variable electorate. *Econometrica : Journal of the Econometric Society*, pages 1027–1041, 1973. *Cité page 37.*

Olivier Sobrie, Vincent Mousseau, and Marc Pirlot. Learning a majority rule model from large sets of assignment examples. In *Proceedings of ADT'13*, pages 336–350, 2013. *Cité page 108.*

Olivier Sobrie, Vincent Mousseau, and Marc Pirlot. Learning the parameters of a non compensatory sorting model. In *Proceedings of ADT'15*, pages 153–170, 2015. *Cité page 108.*

Olivier Sobrie, Nicolas Gillis, Vincent Mousseau, and Marc Pirlot. UTA-poly and UTA-splines : additive value functions with polynomial marginals. *European Journal of Operational Research*, 2017a. *Cité page 52.*

Olivier Sobrie, Vincent Mousseau, and Marc Pirlot. A population-based algorithm for learning a majority rule sorting model with coalitional veto. In *Evolutionary Multi-Criterion Optimization*, pages 575–589, 2017b. Proceedings of the 9th International Conference, EMO 2017. *Cité page 108.*

F. Sourd and O. Spanjaard. A multiobjective branch-and-bound framework : Application to the biobjective spanning tree problem. *INFORMS Journal on Computing*, 20(3) :472–484, 2008. *Cité page 154.*

Horst W. Hamacher Stefan Ruzika. Algorithmics of large and complex networks. chapter A Survey on Multiple Objective Minimum Spanning Tree Problems, pages 104–116. Springer-Verlag, Berlin, Heidelberg, 2009. *Cité page 154.*

Ralph E. Steuer. Multiple criteria optimization : theory, computation, and application. *Willey, New York*, 1986. *Cité page 51.*

Bradley S. Stewart and Chelsea C. White III. Multiobjective A*. *Journal of ACM*, 38(4) :775–814, 1991. *Cité pages 122, 124, et 126.*

Ali Fallah Tehrani, Weiwei Cheng, Krzysztof Dembczynski, and Eyke Hüllermeier. Learning monotone nonlinear models using the Choquet integral. *Machine Learning*, 89(1-2) :183–211, 2012. *Cité page 72.*

Mikhail Timonin. Robust optimization of the Choquet integral. *Fuzzy Sets and Systems*, 213(1) :27–46, 2013. *Cité page 67.*

Mikhail Timonin. Axiomatization of the choquet integral for 2-dimensional heterogeneous product sets. *http ://arxiv.org/abs/1507.04167*, 2016. *Cité page 27.*

Vincenç Torra. The weighted OWA operator. *International Journal of Intelligent Systems*, 12(2) : 153–166, 1997. *Cité pages 11, 26, 28, et 248.*

George Tsaggouris and Christos Zaroliagis. Multiobjective optimization : Improved fptas for shortest paths and non-linear objectives with applications. *Theory of Computing Systems*, 45(1) :162–186, 2009a. *Cité page 142.*

George Tsaggouris and Christos D. Zaroliagis. Multiobjective optimization : Improved FPTAS for shortest paths and non-linear objectives with applications. *Theory Comput. Syst.*, 45(1) :162–186, 2009b. *Cité page 12.*

Amos Tversky and Daniel Kahneman. Judgment under uncertainty : Heuristics and biases. *Sciences*, 185(4175) :1124–1131, 1975. *Cité page 51.*

Kazuki Uematsu and Yoonkyung Lee. Statistical optimality in multipartite ranking and ordinal regression. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(5) :1080–1094, 2015. *Cité page 94.*

Daniel Vanderpooten. The interactive approach in mcda : A technical framework and some basic conceptions. *Mathematical and Computer Modelling*, 12(10) :1213–1220, 1989. *Cité page 51.*

Daniel Vanderpooten and Philippe Vincke. Description and analysis of some representative interactive multicriteria procedures. *Appl. Math. Comp.*, 83(2-3) :261–280, 1997. *Cité pages 13, 51, 122, et 123.*

Paolo Viappiani and Craig Boutilier. Optimal set recommendations based on regret. In *The 7th International Workshop on Intelligent Techniques for Web Personalization and Recommender Systems (ITWP)*, 2009a. *Cité page 64.*

Paolo Viappiani and Craig Boutilier. Regret-based optimal recommendation sets in conversational recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 101–108. ACM, 2009b. *Cité pages 58 et 89.*

Paolo Viappiani and Craig Boutilier. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Advances in Neural Information Processing Systems*, pages 2352–2360, 2010. *Cité page 53.*

William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1) :8–37, 1961. *Cité page 249.*

Philippe Vincke. *L'aide multicritère à la décision.* Editions de l'Université de Bruxelles - Editions Ellispses, Bruxelles, 1989. *Cité page 17.*

John von Neumann and Oskar Morgenstern. *Theory of games and economic behaviour.* Princeton University Press, 1947. *Cité pages 12, 13, 45, 47, et 218.*

Detlof von Winterfeldt. Structuring decision problems for decision analysis. *Acta Psychologica*, 45(1-3) : 71–93, 1980. *Cité page 16.*

Tianhan Wang and Craig Boutilier. Incremental Utility Elicitation with the Minimax Regret Decision Criterion. In *Proceedings of IJCAI'03*, pages 309–316, 2003. *Cité pages 12, 54, 59, 89, 122, 230, et 233.*

Arthur Warburton. Approximation of pareto optima in multiple-objective, shortest-path problems. *Operations Research*, 35(1) :70–79, 1987. *Cité page 142.*

Robert J. Weber. Comparison of voting. *Ronéotypé*, 1977. *Cité page 39.*

Paul Weng and Bruno Zanuttini. Interactive Value Iteration for Markov Decision Processes with Unknown Rewards. In *Proceedings of IJCAI'13*, pages 2415–2421, 2013. *Cité pages 13 et 122.*

John A. Weymark. Generalized Gini inequality indices. *Mathematical Social Sciences*, 1(4) :409–430, 1981. *Cité page 25.*

John A Weymark. A reconsideration of the harsanyi–sen debate on utilitarianism. *Interpersonal comparisons of well-being*, 255, 1991. *Cité page 218.*

Chelsea C. White III, Andrew P. Sage, and Shigeru Dozono. A model of multiattribute decisionmaking and trade-off weight determination under uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics*, 14(2) :223–229, 1984. *Cité pages 12, 54, 55, 58, et 122.*

Andrzej P. Wierzbicki. On the completeness and constructiveness of parametric characterizations to vector optimization problems. *Operations-Research-Spektrum*, 8(2) :73–87, 1986. *Cité pages 11 et 51.*

Nic Wilson, Abdul Razak, and Radu Marinescu. Computing possibly optimal solutions for multi-objective constraint optimisation with tradeoffs. In *Proceedings of IJCAI'15*, pages 815–822, 2015. *Cité page 122.*

Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009. *Cité page 214.*

Feng Wu, Shlomo Zilberstein, and Xiaoping Chen. Online planning for multi-agent systems with bounded communication. *Artificial Intelligence*, 175(2) :487–511, 2011. *Cité page 214.*

Lirong Xia and Vincent Conitzer. Determining possible and necessary winners given partial orders. *Journal of Artificial Intelligence Research (JAIR)*, 41 :25–67, 2011. *Cité pages 122, 170, et 193.*

Lirong Xia, Vincent Conitzer, and Jérôme Lang. Strategic sequential voting in multi-issue domains and multiple-election paradoxes. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 179–188, 2011. *Cité page 170.*

Ronald R. Yager. On Ordered Weighted Averaging aggregation operators in multicriteria decision making. *IEEE Transactions on systems, Man, and Cybernetics*, 18(1) :183–190, 1988. *Cité pages 11, 24, 25, et 28.*

H Peyton Young. An axiomatization of borda's rule. *Journal of economic theory*, 9(1) :43–52, 1974. *Cité page 37.*

Lotfi Asker Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy sets and systems*, 1(1) :3–28, 1978. *Cité page 41.*

Gengui Zhou and Mitsuo Gen. Genetic algorithm approach on multi-criteria minimum spanning tree problem. *European Journal of Operational Research*, 114(1) :141–152, 1999. *Cité page 155.*

Stanley Zionts and Jyrki Wallenius. An interactive programming method for solving the multiple criteria problem. *Management Science*, 22(6) :652–663, 1976. *Cité pages 51 et 123.*

Constantin Zopounidis and Michael Doumpos. A multicriteria decision aid methodology for sorting decision problems : The case of financial distress. *Computational Economics*, 14(3) :197–218, 1999. *Cité page 94.*

Constantin Zopounidis and Michael Doumpos. Multicriteria classification and sorting methods : A literature review. *European Journal of Operational Research*, 138(2) :229–246, 2002. *Cité pages 94 et 105.*

Résumé. Les travaux menés dans cette thèse s'inscrivent dans le cadre de la théorie de la décision algorithmique, domaine de recherche à la croisée de la théorie de la décision, de la recherche opérationnelle et de l'intelligence artificielle. Notre objectif dans cette thèse est de concevoir des algorithmes efficaces pour la résolution de problèmes de décision dans des environnements complexes (multicritère, multi-agents, incertain). Nous nous intéressons d'une part à l'élicitation des préférences fondée sur des modèles décisionnels et d'autre part à l'exploitation de ces préférences pour la recherche des solutions optimales sur des espaces définis de manière explicite ou implicite (optimisation combinatoire). Pour la résolution de problèmes combinatoires, nous proposons et étudions une nouvelle approche, consistant à combiner l'élicitation incrémentale des préférences et l'exploration implicite des solutions potentielles. L'intuition sous-jacente est d'utiliser l'exploration des solutions potentielles pour identifier des questions informatives tout en exploitant les réponses obtenues pour mieux focaliser la recherche sur les solutions préférées. Cette approche nous a conduit à proposer des procédures de décision par élicitation incrémentale pour les problèmes de recherche dans un graphe d'états multi-objectifs, les problèmes de chemins optimaux et d'arbre couvrants dans les graphes multicritères, les problèmes de sac à dos multi-agents et les problèmes de décision séquentielle dans l'incertain. Nous établissons des résultats théoriques garantissant la correction des algorithmes proposés et présentons des tests numériques montrant leur efficacité pratique.

Mots-clés : Aide à la décision, élicitation de préférences, optimisation combinatoire, décision multicritère, décision collective, décision dans l'incertain.

Abstract. This thesis work falls within the area of algorithmic decision theory which is at the junction of decision theory, operations research and artificial intelligence. Our aim is to produce algorithms allowing the fast resolution of decision problems in complex environments (multiple criteria, multi-agents, uncertainty). This work focuses on decision-theoretic elicitation and uses preferences to efficiently determine the best solutions among a set of alternatives explicitly or implicitly defined (combinatorial optimization). For combinatorial optimization problems, we propose and study a new approach consisting in interleaving incremental preference elicitation and preference-based search. The idea is to use the exploration to identify informative preference queries while exploiting answers to better focus the search on the preferred solutions. This approach leads us to propose incremental elicitation procedures for multi-objective state-space search problems, multicriteria shortest path problems, multicriteria minimum spanning tree problems, multi-agents knapsack problems and sequential decision problems under uncertainty. We provide theoretical guarantees on the correctness of the proposed algorithms and we present numerical tests showing their practical efficiency.

Keywords : Decision Aid, Preference Elicitation, Combinatorial Optimization, Multicriteria Decision Making, Collective Decision Making, Decision Making Under Uncertainty.