Planification avec préférences basée sur la Théorie de l'Utilité Multi-Attribut couplée à une intégrale de Choquet : application à l'interopérabilité des organisations en gestion de crise

Loïc Bidoux

Université Fédérale

Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

École Nationale Supérieure des Mines d'Albi-Carmaux conjointement avec l'INSA de Toulouse

---

Présentée et soutenue par :
Loïc Bidoux
le mardi 14 juin 2016

Titre :

Planification avec préférences basée sur la Théorie de l'Utilité Multi-Attribut couplée à une intégrale de Choquet : application à l'interopérabilité des organisations en gestion de crise

---

École doctorale et discipline ou spécialité :
EDSYS : Génie Industriel 4200046

Unité de recherche :
Centre Génie Industriel - Ecole des Mines d'Albi-Carmaux

Directeur/trice(s) de Thèse :

Frédérick BENABEN

Jury :

Frédérick BENABEN - Maître Assistant HDR, Ecole des Mines d'Albi-Carmaux - Directeur
Khaled HADJ-HAMOU - Maître de Conférence, Institut Polytechnique Grenoble - Rapporteur
Jacques LAMOTHE - Professeur, Ecole des Mines d'Albi-Carmaux - Co-directeur
Jacky MONTMAIN - Professeur, Ecole des Mines d'Alès - Rapporteur
Jean-Paul PIGNON - Ingénieur spécialiste études amont, Thales - Encadrant
Bartel VAN DE WALLE - Professeur, Université de technologie de Delft - Président

Remerciements

Je tiens à remercier Khaled Hadj-Hamou, Jacky Montmain et Bartel Van De Walle d'avoir accepter de faire partie de mon jury de thèse. La pertinence de leurs remarques couplée à leur gentillesse ont contribué à faire de ma soutenance de thèse un beau moment d'échange scientifique.

Je remercie chaleureusement Frédérick Bénaben pour la confiance dont il a fait preuve à mon égard (laquelle m'a offert l'opportunité de réaliser ces travaux) ainsi que Jacques Lamothe, Jean-Paul Pignon et Sébastien Truptil pour leur encadrement tout au long de ces trois années de thèse. Ils m'ont fourni un cadre de travail - qu'il soit Albigeois ou Parisien - agréable me permettant de réaliser cette étude dans les meilleures conditions possibles. Par ailleurs, leurs expériences et points de vue respectifs ont forgé à jamais ma vision de la recherche contribuant ainsi à faire de moi le scientifique que je suis aujourd'hui. Je remercie une seconde fois Jean-Paul qui en plus d'avoir été un mentor exemplaire est devenu un ami sincère au fur et à mesure que nous apprenions à nous connaître.

Je remercie également Xavier Lagrenade de m'avoir accueilli au sein de son équipe chez Thales. Je remercie Dominique Attali pour sa bienveillance au cours de ces trois dernières années ainsi que l'ensemble de mes collègues Christophe, Eric, Gilles, Jean-Pierre, Marc, Patrice, Philippe, Sonia, Thierry et tous les membres d'XPS pour leur professionnalisme et leur gentillesse au quotidien. J'ai beaucoup appris à leurs côtés notamment lors de conversations dont les thématiques dépassaient (malheureusement trop souvent) mon domaine d'expertise.

Je souhaite aussi remercier l'ensemble de mes collègues de l'Ecole des Mines d'Albi à savoir Didier, Elise, Franck, Isabelle, Lionel, Matthieu, Michel, Paul ainsi que tous les (post)-doctorants du centre pour leur bonne humeur quotidienne, laquelle

m'a permis d'apprécier l'ensemble de mes séjours au laboratoire. Impossible de ne pas mentionner plus particulièrement Alexandre, Anne-Marie, Aurélie, Guillaume, Mylène, Nicolas, Sébastien et Tiexin aux cotés desquels j'ai passé de grands moments, toujours agréables bien que parfois un peu stupides (ce qui semble néanmoins conférer à ces derniers un caractère mémorable certain).

J'adresse évidemment un grand remerciement à toute ma famille ainsi qu'à Amélie, Manon, Morgan, Nathanaël mais aussi à Solène et ses proches. Merci à eux de m'avoir supporté (quelque soit l'acception du terme considérée) lors de ces trois dernières années (et les précédentes). Aucun mot ne saurait décrire avec justesse toute l'importance qu'ils ont eu dans ma vie.

Pour finir, je suis reconnaissant envers tous ceux qui ont été oubliés dans les quelques paragraphes précédents à cause de ma maladresse habituelle et qui, à ne pas en douter, me pardonneront cet écart.

Table des matières

Liste des figures

Chapitre 3

Chapitre 4

Liste des tableaux

Chapitre 1

Chapitre 2

Chapitre 3

Chapitre 4

Liste des algorithmes

Chapitre 3

Chapitre 4

Notations

$\mathbb{R}$	Ensemble des réels
$\mathfrak{P}(E)$	Ensemble des parties de E
$\Sigma = (S, A, E, \gamma)$	Système à événements discrets
S	Ensemble des états
A	Ensemble des actions
E	Ensemble des événements
γ	Fonction de transition d'états
$PC = (\Sigma, s_0, S_G)$	Problème de planification classique
$PT = (\Sigma, s_0, S_{G'})$	Problème de planification avec objectifs étendus
$PP = (PT, \succsim)$	Problème de planification avec préférences
s_0	Formule qui décrit l'état initial d'un problème
G	Formule qui décrit les objectifs restreints d'un problème
G'	Formule qui décrit les objectifs étendus d'un problème
$\succsim$	Relation de préférences
X	Ensemble des solutions d'un problème de planification
$\langle \hat{a}_1, \cdots, \hat{a}_n \rangle$	Plan de longueur n
I_C	Instance d'un problème de planification (classique)
I_N	Instance d'un problème de planification (numérique)
I_P	Instance d'un problème de planification (préférences)
I_M	Instance d'un problème de planification (MAUT)

dim	Dimension d'un instance d'un problème de planification
Atm_I	Ensemble des atomes de I
PNE_I	Ensemble des expressions numériques primitives de I
$\mathcal{N}(\phi)[V]$	Forme normalisée de la formule ϕ par rapport à V
SG	Ensemble de préférences
MC	Ensemble de critères MAUT
FA	Ensemble des actions aplanies
GA	Ensemble des actions closes
$\hat{a}$	Nom de l'action a
Pre_a	Formule qui décrit les préconditions de l'action a
Eff_a	Formule qui décrit les effets de a
Add_a	Formule qui décrit les effets positifs de l'action a
Del_a	Formule qui décrit les effets négatifs de l'action a
NEf_a	Formule qui décrit les effets numériques de l'action a
p	Nombre d'attributs
$P = \{1, \cdots, p\}$	Ensemble des attributs
X	Ensemble des alternatives
X_A	Ensemble des alternatives binaires
$\Omega_k \subset \mathbb{R}$	Espace de définition de l'attribut $k \in P$
$\Omega = \Omega_1 \times \cdots \times \Omega_p$	Espace des attributs
$z_k : X \to \Omega_k$	Fonction attribut de l'attribut $k \in P$
$z : X \to \Omega$	Fonction attribut
$Y = \{z(x) \mid x \in X\}$	Ensemble des alternatives dans l'espace des attributs
$\succsim_D$	Relation de préférence du décideur
$\xi \subset \mathbb{R}$	Échelle de satisfaction commune
$\mathbf{0}_k \in \Omega_k$	Élément totalement insatisfaisant pour le critère k
$\mathbf{1}_k \in \Omega_k$	Élément parfaitement satisfaisant pour le critère k
$u_k : \Omega_k \to \xi$	Fonction d'utilité partielle de l'attribut k
$\psi : \xi^p \to \mathbb{R}$	Fonction d'agrégation de critères

$U : \Omega \to \mathbb{R}$	Fonction d'utilité
$S_w : \mathbb{R}^p \to \mathbb{R}$	Somme pondérée de vecteur de poids w
$\mu : \mathfrak{P}(P) \to [0, 1]$	Fonction de capacité
$m : \mathfrak{P}(P) \to \mathbb{R}$	Transformation de Möbius
$C_\mu : \mathbb{R}_+^p \to \mathbb{R}_+$	Intégrale de Choquet
$\phi(k)$	Valeur de Shapley du critère k
$I(A)$	Indice d'interactions des critères $k \in A$
$\texttt{FF}(I_M, s, G)$	Heuristique Fast-Forward
$\Delta_1(I_M, s)$	Estimation de s par rapport aux objectifs
$\Lambda_1(I_M, s)$	Estimation de s par rapport aux préférences
$\Lambda_2(I_m, s)$	Estimation de s par rapport aux préférences
$h_1(I_M, s)$	Heuristique h_1 de ChoPlan
$h_2(I_M, s)$	Heuristique h_2 de ChoPlan
$h_3(I_M, s)$	Heuristique h_3 de ChoPlan
SR_1	Stratégie de recherche 1 de ChoPlan
SR_2	Stratégie de recherche 2 de ChoPlan
$\succ_P$	Dominance au sens de Pareto

Introduction générale

Les travaux présentés dans ce manuscrit de thèse s'intéressent à la problématique scientifique de la *planification avec préférences basée sur la Théorie de l'Utilité Multi-Attribut (*MAUT*) couplée à une intégrale de Choquet*. De plus, ils s'intéressent également à la problématique sociétale de l'*interopérabilité des organisations lors de la gestion de situations de crise*. Cette introduction générale souligne les liens entre ces deux problématiques et précise en quoi l'approche scientifique retenue permet d'adresser le problème sociétal considéré. En outre, cette introduction propose une grille de lecture qui explicite la structure de ce manuscrit.

De la planification avec préférences à l'interopérabilité des organisations

La planification est le processus cognitif qui précède l'action. Elle consiste à choisir et séquencer un ensemble d'actions par anticipation de leurs résultats afin d'atteindre le mieux possible des objectifs fixés [66]. La plupart des actions du quotidien sont suffisamment simples ou intuitives pour ne pas devoir être planifiées explicitement. Néanmoins face à des situations inhabituelles ou à des tâches complexes à réaliser (problèmes intractables, nombreuses contraintes à considérer, pluralité d'acteurs impliqués...), il peut s'avérer nécessaire, voire indispensable, de planifier avant d'agir. De plus, pour résoudre les problèmes de planification les plus complexes (par exemple ceux adressés par la communauté scientifique de la *planification automatisée*), il est nécessaire de recourir à la puissance de calcul d'une machine. Les techniques de planification automatisée peuvent être utilisées pour résoudre de nombreux problèmes comme par exemple le déplacement de robots autonomes, la gestion des pistes de décollage d'un aéroport, l'organisation d'un projet industriel, le choix de l'itinéraire proposé par un GPS ou encore la création de plans d'action pour la gestion de crise.

La majorité des problèmes de planification admettent plusieurs plans solutions. En conséquence, deux stratégies de résolution sont envisageables. La première consiste à se contenter de n'importe quelle solution (résolution d'un *problème de satisfaisabilité*) tandis que la seconde consiste à chercher une bonne solution voire la meilleure au regard d'un ensemble de *préférences* (résolution d'un *problème d'optimisation*). Une préférence représente un élément de la solution que le décideur souhaiterait voir satisfait dans la mesure du possible sans que sa violation ne remette en cause la validité de la solution considérée. Les préférences peuvent être vues à la fois comme des contraintes faibles du problème et comme des points de vue selon lesquels les solutions sont évaluées les unes par rapport aux autres. En reprenant l'exemple de l'itinéraire de GPS précédemment mentionné, une préférence pourrait représenter le souhait d'un décideur de passer par un ville spécifique, d'emprunter l'autoroute ou encore de privilégier les trajets à faible coût.

L'originalité de cette étude consiste à utiliser un formalisme issu du domaine de l'aide à la décision multicritère (à savoir un modèle MAUT couplé à une intégrale de Choquet) pour modéliser les préférences du/des décideurs. Ce formalisme, qui se base sur des préférences floues, est particulièrement intéressant puisqu'il peut être utilisé pour améliorer le pouvoir expressif du langage formel de référence de la planification avec préférences et permet de représenter avec précision les préférences des décideurs. De plus, des techniques issues du domaine de l'aide à la décision permettent de modéliser des préférences selon ce formalisme uniquement à partir d'informations préférentielles opérationnelles. Par conséquent, des experts opérationnels peuvent représenter leurs préférences sans avoir à renseigner explicitement les paramètres du modèle mathématique sous-jacent.

Les travaux réalisés ont une portée générique et peuvent être utilisés pour résoudre n'importe quel problème de planification avec préférences. Ils sont particulièrement indiqués dans les cas où les préférences du problème peuvent être complexes à appréhender. Cette étude a été appliquée à la problématique de la gestion de crise dans le domaine de la sécurité des citoyens (sécurité civile, catastrophes naturelles, accidents industriels...). Lorsqu'une telle crise survient, de nombreuses organisations (services publics tels que les pompiers ou la gendarmerie, associations, entreprises...) sont mobilisées pour y apporter une réponse collective. Le management de cette réponse est généralement placé sous la responsabilité d'une cellule de crise composée

des différentes parties prenantes de la gestion de crise : un ou plusieurs décideurs assistés par des représentants des partenaires mobilisés et des experts pertinents au vu de la situation. Dans la plupart des cas, les organisations mobilisées sont relativement hétérogènes (au niveau culturel, fonctionnel et technologique) et peu, voire pas du tout, entrainées à travailler ensemble. Ceci génère inévitablement des problèmes de collaboration (définition des objectifs difficile, partage d'informations incomplet, mauvaise coordination des acteurs...) qui peuvent grandement limiter l'efficacité des actions entreprises sur le terrain ; et ce malgré la compétence certaine et indéniable des différents partenaires mobilisés. En cela, les résultats de cette étude peuvent être utilisés pour construire des plans d'action collaboratifs de gestion de crise qui sont susceptibles d'adresser ces difficultés.

La problématique de la gestion de crise intéresse à la fois la société Thales Communications and Security et le laboratoire de Génie Industriel de l'Ecole des Mines d'Albi qui collaborent, plus ou moins étroitement, sur ce sujet depuis 2006. Cette collaboration a été initialisée dans le cadre du projet ANR ISyCri (Interopérabilité des Systèmes en situation de Crise) [137, 139] puis s'est poursuivie par l'intermédiaire de ces travaux de thèse et du projet européen DRIVER (FP7/607798).

Les trois questions présentées ci-dessous sont étudiées dans ce manuscrit. Les réponses qui y sont apportées constituent les contributions scientifiques de cette étude comme mentionné sur la figure 2.

Comment utiliser le modèle MAUT *avec une intégrale de Choquet pour améliorer le pouvoir expressif du langage formel utilisé en planification avec préférences ?*

Comment résoudre des problèmes de planification avec préférences formalisés à l'aide d'un modèle MAUT *et d'une intégrale de Choquet ?*

Comment construire des plans d'action collaboratifs pour supporter l'interopérabilité d'un ensemble d'organisations lors de la gestion d'une crise ?

Grille de lecture du manuscrit

Ce manuscrit est structuré selon une approche de type « du plus générique au particulier ». En conséquence, les chapitres 1, 2 et 3 ont été rédigés sans considération particulière pour le domaine de la gestion de crise alors que le chapitre 4 lui est entièrement consacré. De manière à cependant guider le lecteur sensible à cette problématique applicative, des paragraphes spécifiquement dédiés au domaine de la gestion de crise jalonnent les trois premiers chapitres. Ces derniers sont identifiables par leur mise en forme particulière (cf. figure 1).

Domaine de la gestion de crise
...

FIGURE 1 – Mise en exergue d'éléments liés au domaine de la gestion de crise

La figure 2 présente les relations entre les différentes sections de ce manuscrit et précise leurs rôles respectifs. Les chapitres 1 et 2 constituent des états de l'art mis au service des chapitres 3 et 4 ; lesquels exposent les contributions et les résultats de cette étude.

Le chapitre 1 est un état de l'art qui présente les problèmes de la planification dite classique (section 1.1) et de la planification avec préférences (section 1.2). Ces deux problèmes sont introduits à l'aide d'un exemple illustratif (inspiré de l'un des problèmes de référence utilisé par la communauté de la planification) avant d'être décrits formellement.

Le chapitre 2 est également un état de l'art. Il s'intéresse à l'un des formalismes employés en aide à la décision multicritère pour modéliser des préférences. Après avoir brièvement présenté le domaine de l'aide à la décision multicritère (section 2.1), il introduit la Théorie de l'Utilité Multi-Attribut (section 2.2) ainsi que l'intégrale de Choquet (section 2.3). Pour finir, une méthode permettant de construire des modèles de préférences sur la base de ces éléments est décrite (section 2.4).

Le chapitre 3 propose quant à lui une approche originale pour la planification avec préférences qui s'appuie sur les éléments du domaine de l'aide à la décision multicritère présentés dans le chapitre 2. Il s'intéresse dans un premier temps au

gain de pouvoir expressif lié à l'utilisation d'un modèle MAUT et d'une intégrale de Choquet pour représenter les préférences d'un problème de planification (section 3.1). Un algorithme et des heuristiques permettant de résoudre ce type de problèmes sont ensuite proposés (section 3.2). Ces résultats ont conduit à l'implémentation du planificateur CHOPLAN dont les performances ont été comparées à celle des planificateurs de l'art (section 3.3).

Enfin, le chapitre 4 s'intéresse à la question de l'interopérabilité des organisations lors de la réponse à une crise. Après avoir introduit les enjeux associés à cette problématique (section 4.1), il présente une démarche de construction de plans d'action collaboratifs (section 4.2). Cette dernière a été utilisée pour implémenter un prototype logiciel dont le fonctionnement est illustré à l'aide d'un exemple qui met en œuvre un scénario de grandes inondations en Europe du Nord (section 4.3).

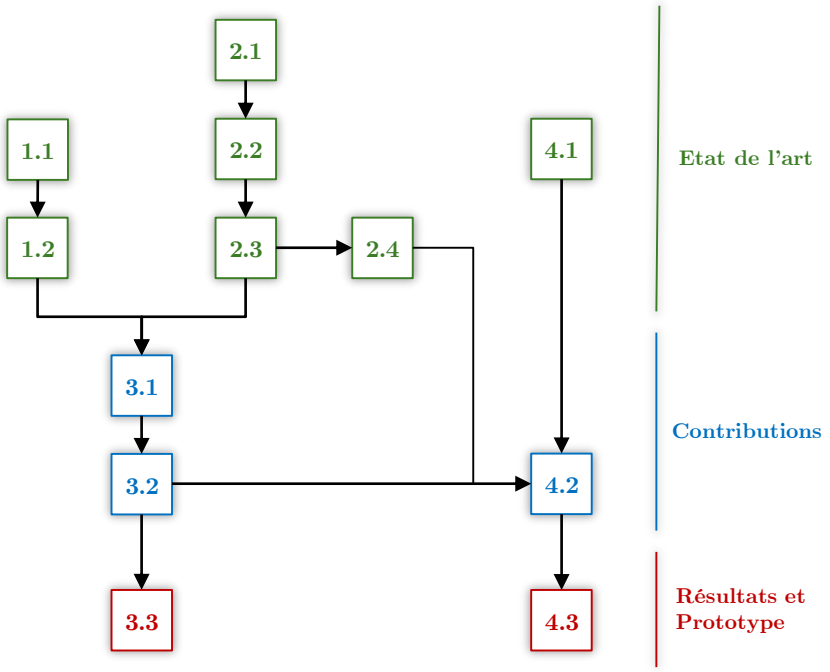


FIGURE 2 – Relations entre les différentes sections de ce manuscrit

Chapitre 1

Planification avec préférences

Ce chapitre s'intéresse au problème de la *planification* qui consiste à élaborer des plans d'action afin d'atteindre des objectifs préalablement définis. Il se focalise principalement sur la problématique de la *planification avec préférences* dont le but est de produire des plans qui maximisent la satisfaction des décideurs. En conséquence, ce chapitre propose une introduction à la planification (cf. section 1.1) ainsi qu'un état de l'art du domaine de la planification avec préférences (cf. section 1.2).

Les deux parties qui composent ce chapitre sont construites symétriquement de sorte à se répondre l'une à l'autre. Ainsi, la section 1.1.1 présente les principaux concepts de la planification tandis que la section 1.2.1 traite de la notion de préférence. De même, les sections 1.1.2 et 1.2.2 introduisent respectivement le problème de la planification classique et celui de la planification avec préférences. Les sections 1.1.3 et 1.2.3 s'intéressent quant à elles aux langages formels utilisés pour représenter et résoudre les problèmes de planification. Finalement, la section 1.2.4 présente les principaux algorithmes de planification avec préférences de l'art.

1.1 De la problématique de la planification

La planification automatisée est un domaine de l'Intelligence Artificielle qui consiste à choisir et séquencer un ensemble d'actions par anticipation de leurs résultats afin d'atteindre des objectifs fixés [66]. Si cette tâche est souvent triviale pour un certain nombre de problèmes du quotidien, elle demeure néanmoins complexe lorsque les problèmes considérés sont fortement combinatoires. Cette section introduit le modèle conceptuel de la planification (cf. section 1.1.1) ainsi que le problème dit de *planification classique* (cf. section 1.1.2). De plus, le langage formel utilisé pour spécifier et résoudre les problèmes de planification est également présenté (cf. section 1.1.3).

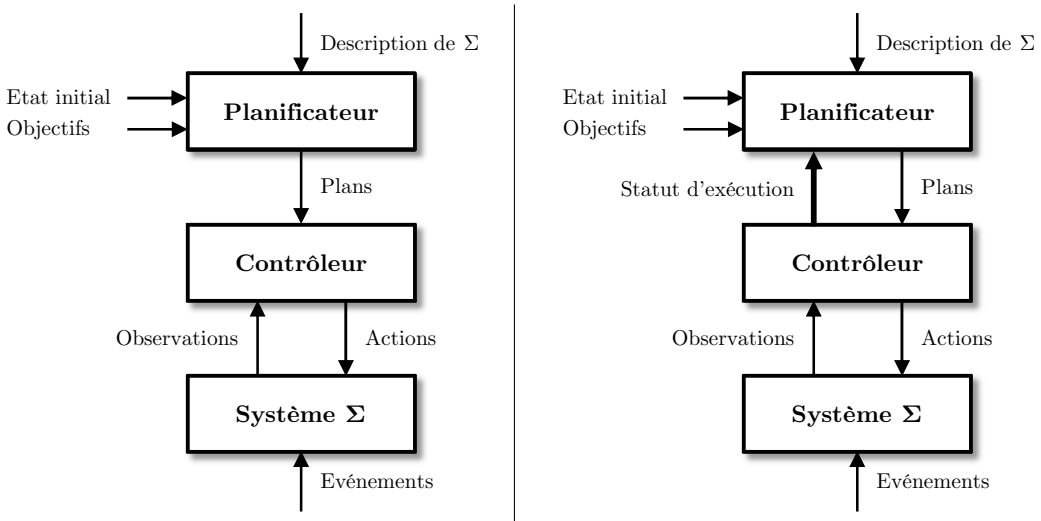
1.1.1 Modèle conceptuel de la planification

La planification a pour objectif de faire évoluer un système Σ dans un état initial connu vers un état final satisfaisant un ensemble d'objectifs fixés. Dans son acception la plus générique, l'activité de la planification englobe à la fois l'élaboration du plan d'action considéré ainsi que l'exécution de ce dernier. Le modèle conceptuel de la planification formalise l'activité de la planification et a été proposé dans l'ouvrage de référence *Automated Planning : Theory and Practice* [66]. Il repose sur les interactions de trois composants : un planificateur, un contrôleur et un système Σ (cf. figure 1.1a).

1. Le planificateur construit le plan d'action en se basant sur la description du système Σ, de son état initial et des objectifs à atteindre.
2. Le contrôleur exécute le plan qui lui est transmis produisant ainsi des actions sur le système. Il s'appuie sur les informations (éventuellement incomplètes) qu'il possède sur l'état actuel du système.
3. Le système Σ évolue en réponse aux actions qui sont exécutées par le contrôleur ou aux événements extérieurs qui surviennent.

Le contrôleur est généralement supposé assez robuste pour gérer les différences qui peuvent exister entre le monde réel et son modèle Σ. Si cette hypothèse n'est pas acceptable, le contrôleur peut retourner un statut d'exécution au planificateur permettant à ce dernier de produire un nouveau plan lorsque la situation observée

et la situation attendue divergent. Le terme de planification dynamique est alors employé puisque la création du plan et son exécution deviennent intimement liées (cf. boucle de rétroaction sur la figure 1.1b).



(a) Version statique (b) Version dynamique

FIGURE 1.1 – Modèle conceptuel de la planification [66]

D'un point de vue formel, le modèle des systèmes à événements discrets [40] est utilisé pour spécifier et capturer l'évolution dynamique du système Σ.

Définition 1.1 - Système à événements discrets [40]
Un système à événements discrets est un quadruplet $\Sigma = (S, A, E, \gamma)$ avec :

- S un ensemble fini ou récursivement énumérable d'états ;
- A un ensemble fini ou récursivement énumérable d'actions ;
- E un ensemble fini ou récursivement énumérable d'événements ;
- $\gamma : S \times A \times E \to \mathfrak{P}(S)$ une fonction de transition d'états.

Modéliser Σ comme un système à événements discrets revient à caractériser ce dernier par l'état $s \in S$ dans lequel il se trouve. Une évolution du système Σ correspond donc à un changement d'état $s \to s'$. Dans le modèle conceptuel de la planification, la transition d'un état s vers un état s' est représentée par l'application d'un couple action/événement (a, e) dans l'état s. L'action neutre α et l'événement

neutre ϵ sont introduits afin de décrire des transitions provoquées uniquement par des actions ou des événements. Dans de tels cas, les notations $\gamma(s, a)$ et $\gamma(s, e)$ sont utilisées en lieu et place de $\gamma(s, a, \epsilon)$ et $\gamma(s, \alpha, e)$. Il convient de remarquer que bien que les actions et les événements contribuent tous deux à l'évolution du système, leur sémantique diffère. Les actions sont contrôlées par les personnes en charge de l'exécution du plan alors que les événements sont des transitions qui échappent à leur contrôle. Ces derniers peuvent être causés par la dynamique interne du système ou encore par l'évolution de l'environnement du système.

La fonction γ précise l'ensemble des états s' vers lesquels le système Σ est susceptible d'évoluer à partir de l'état s en réponse à une action a, un événement e ou un couple action/événement (a, e). Ainsi, la taille de l'ensemble retourné par $\gamma(s, a)$ notée $|\gamma(s, a)|$ fournit plusieurs informations quant à l'exécution de a dans s. Si $|\gamma(s, a)| > 0$, l'action a est dite applicable dans s puisque le système Σ peut évoluer vers au moins un état s' lors de l'exécution de a dans s. De plus, si $|\gamma(s, a)| \leq 1$ alors l'application de a dans s est dite déterministe. En effet, si l'action a est exécutable et qu'elle est exécutée dans s, Σ ne peut alors évoluer que vers un unique état s'.

Le modèle conceptuel de la planification [66] propose également huit hypothèses pour caractériser les différentes classes de problèmes de planification.

Hypothèse H1 (Σ fini). L'ensemble S des états de Σ est fini.

Hypothèse H2 (Σ complètement observable). L'état du système Σ est entièrement observable. Par conséquent, les observations du système sur lesquelles s'appuie le contrôleur sont toutes parfaites.

Hypothèse H3 (Σ déterministe). Le système Σ est déterministe si pour tout état s, pour toute action a et pour tout événement e, $|\gamma(s, a, e)| \leq 1$. Ainsi, en réponse à un couple action/événement (a, e) applicable dans s, le système Σ ne peut évoluer que vers un unique état s'.

Hypothèse H4 (Σ statique). L'ensemble E des événements de Σ est vide.

Hypothèse H5 (Plans séquentiels). Un problème de planification admet des plans solutions représentés par une séquence d'actions finie ordonnée linéairement, il n'y a donc aucune parallélisme dans les plans solutions.

Hypothèse H6 (Objectifs restreints). Les objectifs à atteindre ne portent que sur l'état final du système. Ainsi, une solution est une séquence de transitions d'états qui aboutit à un état final $s \in S_G$ avec S_G l'ensemble des états dans lesquels les objectifs sont vérifiés.

Hypothèse H7 (Temps implicite). Les actions et événements n'ont aucune durée intrinsèque, les transitions sont donc considérées instantanées.

Hypothèse H8 (Planification statique). Aucun mécanisme de planification dynamique n'est mis en œuvre.

1.1.2 Problème de la planification classique

Le terme *planification classique* fait référence à la classe des problèmes obtenue lorsque les huit hypothèses du modèle conceptuel de la planification sont considérées simultanément. L'étude de ce problème est fondamentale puisque la majorité des problèmes de planification sont définis par extension de ce dernier à l'image de la problématique de la planification avec préférences.

Problème de la planification classique [66] *Étant donné un système à événements discrets $\Sigma = (S, A, \gamma)$, un problème de planification classique est défini par le triplet $PC = (\Sigma, s_0, S_G)$ où s_0 est un état initial et $S_G \subset S$ est l'ensemble des états qui vérifient les objectifs G.*

Une solution de PC est une séquence d'actions $\langle a_1, \cdots, a_n \rangle$ qui correspond à une séquence d'états $\langle s_0, \cdots, s_n \rangle$ telle que :

$$s_1 = \gamma(s_0, a_1), \cdots, s_n = \gamma(s_{n-1}, a_n) \quad et \quad s_n \in S_G.$$

Exemple illustratif

Pour illustrer la problématique de la planification classique, une version simplifiée du problème *Rovers* [42] est utilisée. Dans ce problème, des robots réalisent une exploration planétaire. Ils doivent prendre des photographies et récolter des échan-

tillons de sols ou de roches de plusieurs lieux différents. La récolte des échantillons de sols et de roches est modélisée par deux actions distinctes dans ce problème pour préciser que les robots utilisent des outils différents dans les deux cas.

L'exemple considéré ici s'intéresse au cas d'un unique robot $N1$ qui évolue à travers six lieux Li différents. L'ensemble { robot, planète } constitue le système Σ et évolue en fonction des actions A du robot : déplacement d'un lieu à un autre, prise de photographies et récolte d'échantillons. Tous les lieux ne sont pas accessibles les uns des autres en raison de la présence d'obstacles infranchissables par le robot. En outre, pour pouvoir prendre la photographie d'un lieu $L1$, le robot doit soit se situer dans $L1$ soit dans un lieu $L2$ à partir duquel $L1$ est visible. Dans l'état initial s_0 (représenté sur la figure 1.2), le robot est en $L5$, des échantillons de sols sont présents dans tous les lieux mais seuls les lieux $L2$, $L3$ et $L6$ possèdent des échantillons de roches. Enfin, les objectifs G sont notés [S4], [R3] et [P1] et désignent respectivement l'acquisition d'un échantillon de sol de $L4$, l'acquisition d'un échantillon de roche de $L3$ et la prise d'une photographie de $L1$.

Cet exemple sera enrichi (utilisation de plusieurs robots...) tout au long de ce chapitre pour présenter les mécanismes de la planification ainsi que pour illustrer le problème de la planification avec préférences (cf. sections 1.2.2, 1.1.3 et 1.2.3).

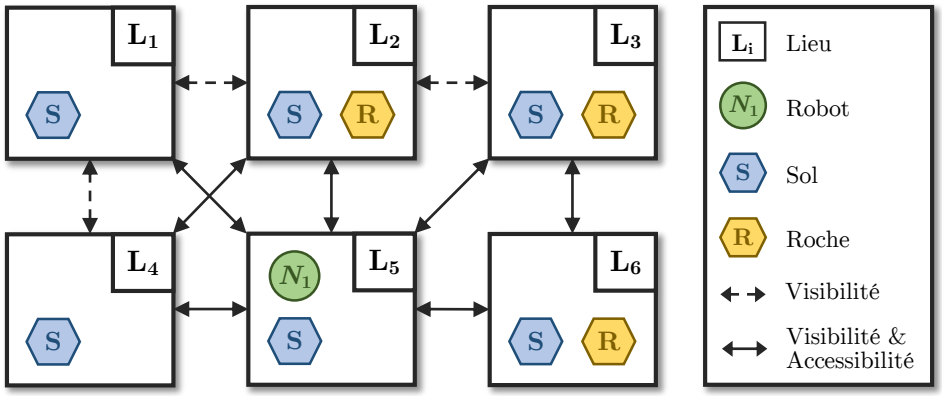


FIGURE 1.2 – Situation initiale de l'exemple *Rovers*

Le plan 1 est l'une des solutions du problème considéré. En effet, la séquence d'actions proposée est valide au regard de la fonction de transition γ et les objectifs [S4], [R3] et [P1] sont tous atteints dans l'état final.

Plan 1
1. *Naviguer avec N1 de L5 vers L3*
2. *Prélever échantillon roche avec N1 en L3*
3. *Naviguer avec N1 de L3 vers L5*
4. *Naviguer avec N1 de L5 vers L4*
5. *Prélever échantillon sol avec N1 en L4*
6. *Photographier L1 avec N1 en L4*

FIGURE 1.3 – Plan 1 : Solution de l'exemple *Rovers*

Du fait de l'objectif d'illustration de cette section, l'exemple retenu a été volontairement choisi de sorte à être très facile à appréhender et à résoudre. Les problèmes de planification sont généralement plus complexes et fortement combinatoires à l'image de celui mentionné dans l'annexe B.5. Pour prendre la mesure de cette complexité, un exemple inspiré des problématiques rencontrées dans les plateformes de transports multimodales est considéré. Dans ce problème, on cherche à déplacer un certain nombre de containers d'un point de stockage à un autre. Les containers peuvent être empilés les uns sur les autres formant ainsi des piles. Chaque point de stockage peut abriter un certain nombre de piles. Pour résoudre le problème, plusieurs robots capables de soulever et transporter des containers sont mis à disposition. Lorsque ce problème est instancié avec cinq points de stockage, trois piles par point de stockage, trois robots et cent containers, le système peut se trouver dans approximativement 10^{277} états différents [66]. A titre de comparaison, l'âge de l'univers est estimé à 10^{17} secondes [19] et l'univers observable serait constitué de 10^{80} atomes [46].

1.1.3 Langage formel pour la planification

Afin de standardiser la représentation des problèmes de planification, le langage PDDL (Planning Domain Definition Language) a été proposé en 1998 par la communauté de la planification [107]. Le PDDL est couramment utilisé et a connu quatre évolutions majeures depuis sa création [49, 61, 64, 91]. Cette section a pour but de présenter en détails les mécanismes de la planification. A cette fin, la syntaxe du PDDL et la sémantique qui y est associée sont introduits. Les éléments de

sémantique présentés traitent de la planification classique et de l'exploitation de variables numériques. En outre, ils sont illustrés à l'aide de l'exemple introduit dans la section 1.1.2.

Syntaxe pour la planification

Le PDDL a été élaboré en s'inspirant de plusieurs travaux antérieurs dont notamment les langages STRIPS [99] et ADL [114], le formalisme UCPOP [11] ainsi que le *situation calculus* [106, 120]. Par ailleurs, le PDDL présente de nombreuses similarités avec la logique du premier ordre. Les notions de *prédicats*, d'*arité*, de *formules*, d'*atomes*, de *littéraux* et de *formules closes* qui sont utilisées dans la suite de cette section sont définies dans l'annexe A qui présente la syntaxe de la logique du premier ordre.

L'ensemble des états caractérisant un problème de planification ne peut généralement pas être énuméré explicitement au vu de sa grande taille. Pour contourner ce problème et représenter de façon compacte les problèmes de planification, les états sont donc caractérisés à l'aide de formules de la logique du premier ordre dont les variables peuvent éventuellement être typées. Ainsi, pour formaliser un problème de planification en PDDL, les principaux éléments à renseigner sont des prédicats et des objets, des actions, une situation initiale et des objectifs et contraintes. La notation BNF [4] est utilisée pour décrire avec précision et sans ambigüité la syntaxe du PDDL [63, 91]. Quelques extraits du langage PDDL sont exposés ci-dessous. Des exemples complets sont disponibles dans l'annexe B.

```
(:objects
  N1 - robot
  L1 - lieu   L2 - lieu   L3 - lieu
  L4 - lieu   L5 - lieu   L6 - lieu
)
```

FIGURE 1.4 – Objets de l'exemple *Rovers*

```
(:predicates
  (presence_echantillon_sol ?x - lieu)
  (presence_echantillon_roche ?x - lieu)
  (visible ?x - lieu ?y - lieu)
  (accessible ?x - lieu ?y - lieu)
  (position ?x - robot ?y - lieu)
  (possede_echantillon_sol ?x - robot ?y - lieu)
  (possede_echantillon_roche ?x - robot ?y - lieu)
  (possede_photographie ?x - robot ?y - lieu)
)
```

FIGURE 1.5 – Prédicats de l'exemple *Rovers*

```
(:action Naviguer
  :parameters (?x - robot ?y - lieu ?z - lieu)
  :precondition (and
    (position ?x ?y) (accessible ?y ?z))
  :effect (and
    (not (position ?x ?y)) (position ?x ?z))
)
```

FIGURE 1.6 – Action Naviguer de l'exemple *Rovers*

```
(:init
  (presence_echantillon_sol L1)
  (presence_echantillon_sol L2)
  (presence_echantillon_roche L2)
  ...
  (visible L1 L2) (visible L2 L1)
  (accessible L1 L5) (accessible L5 L1)
  ...
  (position N1 L5)
)
```

FIGURE 1.7 – Situation initiale de l'exemple *Rovers*

```
(:goal (and
  (possede_echantillon_sol N1 L4)
  (possede_echantillon_roche N1 L3)
  (possede_photographie N1 L1)
))
```

FIGURE 1.8 – Objectifs pour l'exemple *Rovers*

Sémantique pour la planification classique

La syntaxe du PDDL définit l'ensemble des symboles utilisables pour formaliser un problème de planification. Il convient à présent d'introduire la sémantique qui y est associée afin de préciser le sens de ces symboles. Cette dernière permet d'expliquer comment les problèmes de planification sont résolus d'un point de vue conceptuel. En conséquence, elle constitue une étape fondamentale de la résolution des problèmes de planification.

Les définitions présentées sont issues de la version 2.1 du PDDL [61]. Elles ont été simplifiées afin de se limiter aux seuls éléments utilisés dans le cadre de cette étude. En particulier, les aspects temporels formalisés dans [61] n'ont pas été retenus. La notion d'instance d'un problème de planification est présentée par la définition 1.2. Les définitions 1.3 et 1.4 précisent la structure des états S du système Σ considéré. Les définitions 1.5 à 1.9 s'intéressent quant à elles aux actions du problème. Finalement, la notion de plan valide est introduite à l'aide des définitions 1.10 à 1.13.

Définition 1.2 - Instance d'un problème de planification [61]
Une instance d'un problème de planification classique I_C est une paire $(Dom, Prob)$ définie telle que :

- $Dom = (R, A)$ est le *domaine de planification*. Il est constitué d'un ensemble de prédicats R et d'un ensemble d'actions A.
- $Prob = (O, s_0, G)$ est le *problème de planification*. Il est constitué d'un ensemble d'objets O et des formules s_0 et G qui décrivent respectivement l'état initial et les objectifs à atteindre dans l'état final.

L'objet *Dom* constitue une caractérisation générique de l'instance I_C (prédicats considérés et actions réalisables) tandis que *Prob* précise les caractéristiques spécifiques de l'instance I_C (état initial et objectifs du problème). Par conséquent, un domaine de planification peut être utilisé par plusieurs problèmes de planification différents définissant alors autant d'instances de planification.

L'ensemble Atm_{I_C} des atomes de I_C est formé par l'application des prédicats du domaine R aux objets du problème O (en respectant l'arité des prédicats).

Les notions de *structure d'état* et de *valeur de vérité d'une formule dans un état* sont à présent introduites. Considérées conjointement, ces deux définitions permettent de décrire l'ensemble des états dans lequel un système à événements discrets Σ peut se trouver sans pour autant devoir les énumérer explicitement.

Définition 1.3 - Structure d'un état [61]
Étant donné une instance I_C, un état s est défini par la donnée d'un ensemble $Atm \in \mathfrak{P}(Atm_{I_C})$. Atm est appelé l'état logique de s et constitue un sous-ensemble des atomes de I_C.

Définition 1.4 - Valeur de vérité d'une formule [61]
Soit un état $s = (Atm)$, la formule atomique ϕ est vraie dans s si et seulement si $\phi \in Atm$. La notation $s \models \phi$ est utilisée pour dénoter que ϕ est vraie dans s.

La valeur de vérité d'une formule ϕ quelconque (qui peut contenir des connecteurs et quantificateurs logiques comme précisé dans l'annexe A) est déterminée à partir des valeurs de vérité des formules atomiques qui la composent en utilisant l'interprétation usuelle de la logique du premier ordre [37].

Dans l'exemple de la section 1.1.2, le lieu $L4$ est accessible et visible depuis le lieu $L5$ (cf. figure 1.2). Soit $s_0 = (Atm_0)$ l'état initial du problème et ϕ_1, ϕ_2 les formules atomiques qui correspondent aux prédicats PDDL `(accessible L4 L5)` et `(visible L4 L5)`. Les formules ϕ_1 et ϕ_2 sont vraies dans s_0 ($\phi_1, \phi_2 \in Atm_0$) ce qui permet notamment d'écrire que la formule $\phi_3 = \phi_1 \wedge \phi_2$ l'est également ($s_0 \models \phi_3$).

De la même façon que les prédicats sont instanciés avec les objets du problème, les actions d'une instance sont successivement transformées en *actions aplanies* puis en *actions closes* (cf. définition 1.5 à 1.7). Une fois ces transformations effectuées, il devient possible d'introduire les notions d'*applicabilité d'une action* dans un état et de résultat de l'*exécution d'une action* dans un état (voir définitions 1.8 et 1.9).

Définition 1.5 - Structure d'une action [61]
Une action $a \in A$ d'une instance I_C est un triplet ($\hat{a}$, Pre_a, Eff_a) avec :

- $\hat{a}$: le nom de l'action
- Pre_a : une formule qui décrit les préconditions de a
- Eff_a : une formule qui décrit les effets de a

Définition 1.6 - Actions aplanies [61]
Étant donné une instance I_C, l'ensemble des *actions aplanies* FA est défini comme l'ensemble contenant initialement A et construit de la façon suivante :

> Tant que FA contient une action a ayant une formule avec un quantificateur, remplacer a par une version dans laquelle la formule quantifiée (Q (v_1, $\cdots$, v_k) P) est remplacée par la conjonction (si Q est le quantificateur universel) ou la disjonction (si Q est le quantificateur existentiel) des atomes obtenus par toutes les substitutions possibles des objets de I_C dans les variables v_1, $\cdots$, v_k du prédicat P.

Afin d'illustrer le mécanisme de génération des actions aplanies, l'action a_1 `Panorama_planète ?x - robot ?y - lieu` est considérée. Pour que a_1 soit exécutable, le robot doit se trouver dans un lieu à partir duquel tous les lieux de la planète sont visibles. Ainsi, Pre_{a_1} est représenté par la formule :

```
(and (position ?x ?y) (forall (?z - lieu) (visible ?y ?z)))
```

Après aplanissement de a_1, la forme de Pre_{a_1} devient :

```
(and (position ?x ?y) (visible ?y L1) (visible ?y L2)
(visible ?y L3) (visible ?y L4) (visible ?y L5) (visible ?y L6))
```

Définition 1.7 - Actions closes [61]
Soit une instance I_C et une action $a_F \in FA$ formée par aplanissement de $a \in A$. L'ensemble des *actions closes* de a noté GA_a est formé en réalisant l'ensemble des substitutions possibles de variables de a_F par des objets de I_C. De plus, l'ensemble $GA = \bigcup_{a \in A} GA_a$ est l'ensemble des actions closes de I_C.

Par construction (et par définition de la syntaxe du PDDL [63]), si a_G est une action close, alors Eff_{a_G} est une conjonction de littéraux. Une action close peut donc être vue comme un quadruplet ($\hat{a_G}$, Pre_{a_G}, Add_{a_G}, Del_{a_G}) avec :

- $\hat{a_G}$: le nom de l'action et des objets substitués aux variables de a
- $GPre_{a_G}$: les préconditions de a_G représentées par une formule close
- Add_{a_G} : les effets positifs de a_G constitués de l'ensemble des atomes qui sont considérés comme des littéraux positifs dans Eff_{a_G}
- Del_{a_G} : les effets négatifs de a_G constitués de l'ensemble des atomes qui sont considérés comme des littéraux négatifs dans Eff_{a_G}

L'action a_2 `Naviguer ?x - robot ?y - lieu ?z - lieu` présentée sur la figure 1.6 est utilisée pour illustrer la construction des actions closes. Dans le cadre de l'exemple de la section 1.1.2, l'ensemble GA_{a_2} contient 36 actions (1 robot $\times$ 6 lieux $\times$ 6 lieux) dont notamment l'action `Naviguer N1 L3 L4`. L'atome `(position N1 L4)` est l'unique effet positif de cette action de même que l'atome `(position N1 L3)` est l'unique effet négatif de cette dernière.

Définition 1.8 - Applicabilité d'une action [61]
Étant donné une action $a \in GA$, a est applicable dans s si $s \models Pre_a$.

Définition 1.9 - Execution d'une action [61]
Soit un état s et une action $a \in GA$. Si a est applicable dans s, la fonction $\gamma : S \times GA \rightarrow S$ définit l'état résultant de l'exécution de a dans s par :

$$\gamma(s, a) = (Atm \setminus Del_a) \cup Add_a$$

L'action a_3 `Naviguer N1 L5 L3` a pour préconditions $GPre_{a_3} = \phi_4 \wedge \phi_5$ avec `(position N1 L5)` et `(accessible L5 L3)` les prédicats correspondants respectivement à ϕ_4 et ϕ_5. Puisque $s_0 \models \phi_4$ et $s_0 \models \phi_5$, il en résulte que a_3 est exécutable

dans l'état initial s_0. De plus, Del_{a_3} et Add_{a_3} sont respectivement caractérisés par les atomes `(position N1 L5)` et `(position N1 L3)`. Ainsi, l'exécution de a_3 dans s_0 conduit à un état s_1 dans lequel le robot $N1$ n'est plus dans le lieu $L5$ mais dans le lieu $L3$.

A présent que les mécanismes de transition d'états ont été précisés, il est possible de définir la notion de *plan exécutable* et de *plan valide*.

Définition 1.10 - Structure d'un plan [61]
Étant donné une instance I_C, un plan de I_C est une séquence $\langle \hat{a}_1, \cdots, \hat{a}_n \rangle$ telle que pour tout $i \in \{1, \cdots, n\}$, $\hat{a}_i$ est le nom d'une action close.

Définition 1.11 - Trajectoire d'un plan [64]
Étant donné un plan $x = \langle \hat{a}_1, \cdots, \hat{a}_n \rangle$, la séquence d'états $\langle s_0, \cdots, s_n \rangle$ définie telle que $s_1 = \gamma(s_0, a_1)$, $\cdots$, $s_n = \gamma(s_{n-1}, a_n)$ est appelée *trajectoire de* x.

Définition 1.12 - Exécutabilité d'un plan [61]
Soit une instance I_C et un plan $x = \langle \hat{a}_1, \cdots, \hat{a}_n \rangle$ de trajectoire $\langle s_0, \cdots, s_n \rangle$. Si pour tout $i \in \{1, \cdots, n\}$, a_i est applicable dans s_{i-1} alors x est exécutable.

Définition 1.13 - Validité d'un plan [61]
Soit une instance I_C et un plan $x = \langle \hat{a}_1, \cdots, \hat{a}_n \rangle$ de trajectoire $\langle s_0, \cdots, s_n \rangle$. Le plan x est valide s'il est exécutable et si $s_n \models G$.

Le lecteur peut vérifier que le plan 1 présenté dans la section 1.1.2 est valide. En effet, il vérifie les objectifs présentés par la figure 1.8 tout en étant exécutable par rapport à la situation initiale s_0 (voir figures 1.2 et 1.7).

Sémantique pour les expressions numériques

Cette section enrichit la sémantique de la planification classique avec la notion d'expressions numériques. Ces dernières permettent de représenter des problèmes de planification plus complexes et sont notamment utilisées dans le chapitre 3. Les définitions 1.14 à 1.21 complètent la sémantique introduite par les définitions 1.2 à 1.13 en lui ajoutant les éléments liés aux expressions numériques. Pour illustrer

ces nouvelles notions, l'exemple précédent est étendu en considérant plusieurs robots et en tenant compte de la quantité d'énergie qu'ils consomment lors de leurs déplacements. Afin de simplifier le problème, il sera considéré que chaque déplacement consomme 10 unités d'énergie. De plus, un robot est autorisé à se déplacer uniquement s'il a consommé moins de 75 unités d'énergie depuis le début de la mission.

Définition 1.14 - Instance d'un problème de planification [61]
Une instance d'un problème de planification avec expressions numériques I_N est redéfinie par extension de la définition 1.2 en ajoutant un ensemble F de *fonctions numériques* à $Dom = (F, R, A)$.

L'ensemble PNE_{I_N} des *expressions numériques primitives* de I_N est formé par l'application des fonctions F du domaine aux objets O du problème (en respectant l'arité des fonctions). La dimension de I_N notée *dim* est le nombre d'expressions numériques qui peuvent être construites dans I_N et vaut $|PNE_{I_N}|$.

Il convient de ne pas confondre les fonctions de F qui retournent un réel et les fonctions de la logique du premier ordre qui retournent un objet de I_N. De telles fonctions existent en PDDL [91] mais ne sont pas décrites ici puisqu'elles ne sont pas considérées dans le cadre de cette étude.

Pour prendre en compte cette modification, un vecteur de valeurs numériques doit maintenant être associé à chaque état. De plus, les formules peuvent à présent contenir des opérateurs de comparaison ($=$, $\geq$, $\leq$, $>$, $<$, ...).

Définition 1.15 - Structure d'un état [61]
Par extension de la définition 1.3, un état s est une paire (Atm, v) avec $v \in \mathbb{R}_{\perp}^{dim}$ où $\mathbb{R}_{\perp} = \mathbb{R} \cup \{\perp\}$ et $\perp$ désigne une valeur non définie.

Soit deux variables e_1 et e_2 correspondant à l'énergie consommé par les robots $N1$ et $N2$ lors de leurs déplacements. Chaque état s contient à présent un vecteur $v = (e_1, e_2)$ qui précise les valeurs de e_1 et e_2 dans s.

Définition 1.16 - Normalisation d'une formule [61]
Étant donné une instance I_N de dimension dim, une formule ϕ et un vecteur $V \in \mathbb{R}^{dim}$, soit $Ind : PNE_{I_N} \rightarrow \{1, \cdots, dim\}$ une fonction qui associe à toute expression numérique de I_N l'indice d'un élément de V.

La *forme normalisée* de ϕ par rapport à V notée $\mathcal{N}(\phi)[V]$ est obtenue en substituant chaque expression numérique primitive f de ϕ par le réel $V_{Ind(f)}$.

Définition 1.17 - Valeur de vérité d'une formule [61]
Soit un état $s = (Atm, v)$, la valeur de vérité d'une formule ϕ est définie par extension de la définition 1.4 à partir de $\mathcal{N}(\phi)[V := v]$ la forme normalisée de ϕ dans s en utilisant l'interprétation usuelle des opérateurs de comparaison.

Soit `(<= (energy ?x - robot) 75)` la condition supplémentaire de déplacement des robots. Dans le cas de l'action a_3 `Naviguer N1 L5 L3`, cette formule ϕ_6 s'écrit `(<= (energy N1) 75)` où `(energy N1)` représente l'expression numérique primitive qui associe au robot $N1$ sa consommation d'énergie. Ainsi, la forme normalisée de ϕ_6 par rapport à un vecteur $v = (e_1, e_2)$ est `(<= e1 75)` puisque $Ind($`energy N1`$) = 1$. La précondition ϕ_6 est vraie dans un état s si $s \models \mathcal{N}(\phi_6)[v]$ ce qui s'interprète naturellement par $e_1 \leq 75$.

Les actions peuvent modifier les valeurs numériques des états par l'intermédiaire d'*effets numériques* qui sont constitués de l'opérateur d'affectation `assign`, d'une expression numérique primitive notée *lvalue* et d'une *expression numérique* (expression arithmétique quelconque dont les termes sont des réels ou des expressions numériques primitives) notée *rvalue* [61].

Définition 1.18 - Formule d'affectation [61]
La *formule d'affectation* associée à un effet numérique est formée en remplaçant l'opérateur d'affectation `assign` par l'opération arithmétique $=$ et en annotant l'expression numérique primitive *lvalue* d'un symbole « prime ».

La *forme normalisée* d'une formule d'affectation ϕ par rapport à (V', V) notée $\mathcal{N}(\phi)[V', V]$ est obtenue en substituant chaque expression numérique primitive f de ϕ par le réel $V'_{Ind(f)}$ si f est annotée d'un symbole « prime » et par le réel $V_{Ind(f)}$ dans le cas contraire.

A titre d'exemple, soit l'effet numérique `(assign (energy ?x - robot) (+ (energy ?x - robot) 10))` qui représente la consommation d'énergie des robots lors de leurs déplacements. Dans le cas de l'action a_3, l'expression numérique primitive *lvalue* est `(energy N1)` et l'expression numérique *rvalue* est `(+ (energy N1) 10)`. La formule d'affectation ϕ_7 associée à cet effet numérique est `(= (energy N1)' (+ (energy N1) 10))`. Par conséquent, la forme normalisée de ϕ_7 par rapport à (v', v) avec $v' = (e'_1, e'_2)$ et $v = (e_1, e_2)$ s'exprime `(= e1' (+ e1 10))` ce qui s'interprète par l'affectation $e'_1 = e_1 + 10$.

Définition 1.19 - Actions closes [61]
Une action close est redéfinie par extension de la définition 1.7 à l'aide d'un quintuplet ($\hat{a_G}$, Pre_{a_G}, Add_{a_G}, Del_{a_G}, NEf_{a_G}) avec :

- NEf_{a_G} : les effets numériques de a_G constitués de l'ensemble des formules d'affectation associées aux effets numériques de Eff_{a_G}

Définition 1.20 - Applicabilité d'une action [61]
La notion d'applicabilité d'une action $a \in GA$ dans un état s est redéfinie par extension de la définition 1.8 en ajoutant la condition suivante : aucune expression numérique primitive n'est annotée d'un symbole « prime » plus d'une fois lors de la construction des effets numériques de NEf_a.

Définition 1.21 - Execution d'une action [61]
La fonction $\gamma : S \times GA \rightarrow S$ qui précise l'état $s' = (Atm', v')$ résultant de l'exécution d'une action $a \in GA$ dans un état $s = (Atm, v)$ est redéfinie par extension de la définition 1.9 par l'ajout de la fonction NF obtenue par composition des fonctions de l'ensemble $\{ NF_\phi : \mathbb{R}_\perp^{dim} \rightarrow \mathbb{R}_\perp^{dim} \mid \phi \in NEf_a \}$ avec :

- $NF_\phi(v) = v'$ où $v'_{Ind(f)}$ est défini tel que $\mathcal{N}(\phi)[V' := v', V := v]$ soit satisfaite si f est une expression numérique primitive annotée d'un symbole « prime » ; et $v'_{Ind(f)} = v_{Ind(f)}$ dans le cas contraire.

La condition supplémentaire d'applicabilité permet à toutes les affectations de valeurs numériques d'une action d'être consistantes entre elles. Il en résulte que la loi de composition des fonctions est commutative pour l'ensemble des fonctions NF_ϕ ce qui permet de construire NF sans se soucier de l'ordre dans lequel les fonctions NF_ϕ sont appliquées.

La définition 1.21 peut sembler complexe de prime abord mais son interprétation est relativement naturelle. En effet, la fonction NF est définie de sorte que les affectations de variables réalisées coïncident avec l'effet numérique considéré. Ainsi, dans un problème avec deux robots $N1$ et $N2$ dont les consommations d'énergie sont nulles dans l'état initial s_0, cette définition garantit que l'exécution de l'action a_3 (à savoir un déplacement de $N1$) conduit effectivement à un état s_1 dans lequel $e_1 = 10$ et $e_2 = 0$.

A l'exception des considérations sur les expressions numériques, cette première section s'est focalisée sur le problème de la planification classique. Ce dernier est parfois trop limité pour modéliser correctement certains problèmes réels. Dans ce cas, il est nécessaire de considérer des classes de problèmes plus complexes qui ne reposent que sur un sous-ensemble des hypothèses du modèle conceptuel de la planification (cf. section 1.1.1). Ces nouvelles classes de problèmes forment autant de sous-domaines de la planification à l'image de la *planification temporelle* qui ne considère pas H7, de la *planification dans l'incertain* qui ne considère pas H2, H3 et H5 ou encore de la *planification avec préférences* qui ne considère ni H1 ni H6.

Domaine de la gestion de crise

L'élaboration d'une réponse à une situation critique par les membres d'une cellule de crise constitue un problème de planification. En effet, à partir d'une situation initiale connue (la situation de crise), les décideurs se fixent des objectifs puis essayent de déterminer un plan d'action permettant de les atteindre. Le plan solution séquence les différentes actions qui doivent être mises en œuvres par les partenaires mobilisés afin de résoudre la crise. Il précise à ces derniers ce qu'ils doivent faire mais les laisse entièrement autonomes quant à la réalisation des actions qui leurs sont assignées (principe de subsidiarité).

Les définitions proposées s'appuient sur les notions d'objet, d'état, d'action et d'expressions numériques. En gestion de crise, un état représente simplement l'état du monde à un instant donné et constitue donc une description de l'ensemble des éléments qu'il est pertinent de considérer au vu du problème à

résoudre. Par exemple, pour représenter qu'un bâtiment b_0 est en feu au début de la crise, il est possible de créer un objet bâtiment b_0 et un prédicat $(enFeu\ ?b)$ (pouvant caractériser n'importe quel bâtiment). L'atome $(enFeu\ b_0)$ qui est obtenu par combinaison de ces deux éléments appartient alors à l'état initial s_0. L'action « Eteindre incendie » des pompiers pourrait être utilisée pour passer de l'état s_0 à un état s_1 qui ne contiendrait plus l'atome $(enFeu\ b_0)$. Un autre exemple d'action que les pompiers peuvent être amenés à exécuter consiste à inonder une infrastructure en eau propre dans le but de la protéger (métro, bâtiment...) en cas d'inondation. Par ailleurs, les expressions numériques peuvent être employées pour dénombrer les ressources gérées par la cellule de crise à l'image de denrées alimentaires par exemple.

Pour finir, il convient de discuter de la pertinence des hypothèses du modèle conceptuel de la planification pour le domaine de la gestion de crise. L'approche de planification avec préférences retenue dans cette étude (cf. section 1.2) ne considère pas les hypothèses H1 et H6. Par ailleurs, les hypothèses H2, H3 et H4 ne sont pas toujours compatibles avec la gestion de crise puisque le système Σ n'est pas nécessairement observable, déterministe ou statique. Elles peuvent néanmoins être considérées si l'hypothèse H8 est relaxée. En effet, si de nouveaux éléments concernant l'état de Σ sont portés à la connaissance des décideurs lors de l'exécution du plan solution (violation de H2) ou qu'une action ne produit pas les effets attendus (violation de H3) ou encore qu'un événement inattendu survient (violation de H4), il est possible, dans le cas où l'hypothèse H8 est relaxée, de ré-exécuter le processus de planification pour adresser ces problèmes (approche de type planification dynamique, voir section 4.1.2 pour une discussion à ce propos). De plus, les hypothèses H5 et H7 peuvent également être considérées trop restrictives pour le domaine de la gestion de crise. Il est néanmoins possible de les relaxer en s'appuyant sur des mécanismes d'ordonnancement et des mécanismes de planification temporelle (voir discussion section 4.2.3). Ainsi, des plans d'action pour la gestion de crise peuvent être construits en ne considérant que les hypothèses H2, H3 et H4 dans le cas où une démarche de planification dynamique est utilisée. Dans le cadre de cette étude, l'hypothèse H7 est également retenue puisqu'aucun mécanisme de planification temporelle n'est considéré.

1.2 De la problématique de la planification avec préférences

La thématique des préférences a été très étudiée par la communauté de l'Intelligence Artificielle [32, 44, 67, 117, 121]. Il apparait clairement que les préférences d'un décideur peuvent être modélisées avec beaucoup de diversité. En conséquence, un grand nombre de formalismes au pouvoir expressif varié ont été proposés dans la littérature. Le domaine de la planification automatisée ne déroge pas à cette règle et plusieurs approches ont été suggérées pour résoudre les problèmes de planification avec préférences.

La planification avec préférences se distingue de la planification classique par l'utilisation d'objectifs dits étendus. Ces derniers surclassent les objectifs restreints considérés dans le cadre de l'hypothèse H6 et sont présentés dans la section 1.2.1. Le problème de la planification avec préférences est ensuite introduit dans la section 1.2.2. La section 1.2.3 précise quant à elle les divers formalismes utilisés pour représenter ce type de problèmes. Elle se focalise principalement sur la sémantique du PDDL3 qui est le formalisme le plus employé mais présente également quelques langages alternatifs. Pour finir, les méthodes et algorithmes de l'art employés pour résoudre les problèmes de planification avec préférences sont exposés dans la section 1.2.4.

1.2.1 Objectifs étendus et préférences

Les objectifs étendus peuvent être de natures différentes. On distingue les *objectifs* (ou contraintes fortes) qui doivent nécessairement être vérifiés par les solutions du problème des *préférences* (ou contraintes faibles) qui peuvent ne pas être vérifiées par un plan solution.

Il est possible de subdiviser à nouveau les objectifs en deux catégories. Les *objectifs finaux* correspondent exactement aux objectifs restreints de l'hypothèse H6. Des *objectifs de trajectoires* peuvent également être considérés afin de contraindre les séquences d'action solutions d'un problème. Ils permettent par exemple d'interdire les transitions vers certains états spécifiques ou encore de forcer le système à passer par un état avant un autre...

A l'image des objectifs, les préférences peuvent représenter des conditions sur les états finaux (*préférences finales*) ou sur les états intermédiaires (*préférences de trajectoires*) des plans solutions. Néanmoins, elles peuvent également porter sur n'importe quelle variable numérique (*préférences numériques*) définie dans le problème. Ceci leur confère un pouvoir expressif certain puisqu'elles peuvent aussi bien décrire des coûts et des risques à minimiser (coût de production, délai de livraison...) que des quantités ou des gains à maximiser (par exemple un niveau de qualité). Dans le cas où des variables numériques sont utilisées pour décrire les états du système, il devient alors impossible de considérer l'hypothèse H1 du système fini.

Dans le cadre de la planification avec objectifs étendus et préférences, les objectifs constituent des *conditions de satisfaisabilité* que les plans solutions doivent nécessairement vérifier. Les préférences constituent quant à elles des *conditions d'optimalité* que l'utilisateur aimerait voir satisfaites si possible. En conséquence, ces dernières permettent de définir la notion de *qualité* (ou d'*utilité*) d'une solution qui peut être utilisée pour identifier les meilleurs plans parmi l'ensemble des plans valides. En effet, il est possible d'introduire une relation d'ordre $\succsim$ sur l'ensemble des plans solutions afin de comparer ces derniers au sens des préférences.

La sémantique associée à $\succsim$ est la suivante : si $x_1 \succsim x_2$, alors x_1 est au moins aussi préféré que x_2. Plusieurs méthodes peuvent être employées pour définir la relation $\succsim$ et calculer l'utilité des plans. Une méthode simpliste pourrait par exemple considérer qu'un plan x_1 est préféré à un plan x_2 s'il satisfait davantage de préférences que x_2. Ainsi, si s_n^1 et s_n^2 sont les états finaux de x_1 et x_2, alors $x_1 \succsim x_2$ si et seulement si le nombre de préférences satisfaites dans s_n^1 est plus grand que le nombre de préférences satisfaites dans s_n^2. L'une des méthodes les plus couramment utilisée (voir section 1.2.3 pour plus de détails) consiste à définir une fonction d'utilité U qui associe à chaque plan une valeur numérique. Ainsi, lorsque cette dernière doit être maximisée, x_1 est préféré à x_2 si et seulement si $U(x_1) \geq U(x_2)$. La fonction U est généralement choisie de sorte à pondérer les différentes préférences du problème en fonction de leurs importances respectives.

1.2.2 Problème de la planification avec préférences

Ces précisions au sujet des objectifs étendus permettent d'introduire formellement le problème de la planification avec préférences.

Problème de la planification avec préférences [7] *Étant donné un système à événements discrets $\Sigma = (S, A, \gamma)$, soit PT un problème de planification défini par le triplet $(\Sigma, s_0, S_{G'})$ où s_0 est un état initial et $S_{G'}$ l'ensemble des états qui vérifient les objectifs étendus G'.*

Soit X l'ensemble des solutions de PT c'est à dire l'ensemble des séquences d'actions $\langle a_1, \cdots, a_n \rangle$ qui correspondent à une séquence d'états $\langle s_0, \cdots, s_n \rangle$ telle que $s_1 = \gamma(s_0, a_1)$, $\cdots$, $s_n = \gamma(s_{n-1}, a_n)$ avec $s_n \in S_{G'}$.

Soit $\succsim$ une relation d'ordre sur X, un problème de planification avec préférences est défini par le couple $PP = (PT, \succsim)$.

L'ensemble des solutions de PP est l'ensemble X des solutions de PT. De plus, un plan $x \in X$ est une solution optimale si $\forall x' \in X,\ x \succsim x'$.

L'ensemble $\Lambda_P = \{x \in X \mid \forall x' \in X,\ x \succsim x'\}$ tel que défini précédemment est l'ensemble des solutions optimales de PT au sens de $\succsim$. La planification avec préférences constitue donc un problème d'optimisation (recherche d'une des meilleures solutions) et s'oppose en cela à la planification classique qui constitue un problème de satisfaisabilité (recherche d'une solution quelconque).

Exemple illustratif

L'exemple introduit à la section 1.1.2 est à présent enrichi afin de constituer un problème de planification avec préférences. Aux objectifs finaux [S4], [R3] et [P1] vient s'ajouter l'objectif de trajectoire [SometimeBefore R3 S4] qui s'interprète comme la contrainte que l'acquisition d'un échantillon de sol du lieu $L4$ soit réalisée avant l'acquisition d'un échantillon de roche du lieu $L3$. Le plan 1 présenté dans la section 1.1.2 ne respecte pas cette contrainte et ne constitue par conséquent pas une solution du problème nouvellement défini.

Cinq préférences notées [R2], [S6], [Sometime N1 L1], [AtMostOnce N1 L6] et [ConsommationEnergie N1] sont également ajoutées au problème initial. Les préférences [R2] et [S6] sont des préférences finales et s'interprètent respectivement comme le souhait de voir réaliser l'acquisition d'un échantillon de roche du lieu $L2$ et l'acquisition d'un échantillon de sol du lieu $L6$. Les préférences [Sometime N1 L1] et [AtMostOnce N1 L6] sont quant à elles des préférences de trajectoires qui s'interprètent respectivement comme le souhait que le robot $N1$ soit dans le lieu $L1$ dans au moins l'un des états induits par le plan et ne soit pas dans le lieu $L6$ dans plus d'un des états induit par le plan. Si la notion d'expression numérique présentée dans la section 1.1.3 est utilisée, il serait également possible de construire une préférence numérique à partir de la variable [ConsommationEnergie N1]. Cette dernière représenterait alors la satisfaction du décideur quant à la quantité d'énergie consommée par le robot $N1$ lors de l'exécution du plan.

Le plan 2 est une solution du problème puisque sa séquence d'actions est valide et qu'il vérifie les objectifs [S4], [R3], [P1] et [SometimeBefore R3 S4].

Plan 2

1. *Naviguer avec N1 de L5 vers L4*
2. *Prélever échantillon sol avec N1 en L4*
3. *Photographier L1 avec N1 en L4*
4. *Naviguer avec N1 de L4 vers L2*
5. *Prélever échantillon roche avec N1 en L2*
6. *Naviguer avec N1 de L2 vers L5*
7. *Naviguer avec N1 de L5 vers L6*
8. *Prélever échantillon sol avec N1 en L6*
9. *Naviguer avec N1 de L6 vers L3*
10. *Prélever échantillon roche avec N1 en L3*

Le plan 2 satisfait les préférences [R2], [S6] et [AtMostOnce N1 L6] mais ne satisfait pas [Sometime N1 L1] puisque le robot ne visite pas le lieu $L1$. Pour pouvoir comparer le plan 2 à d'autres plans solutions, il serait nécessaire de préciser l'expression de la relation d'ordre $\succsim$ à partir des préférences considérées.

Par ailleurs, il convient de remarquer que la construction de $\succsim$ soulève plusieurs questions. Comment représenter simultanément et de façon cohérente des préférences numériques comme la quantité d'énergie consommée et des préférences non numériques ? Comment gérer les importances relatives des différentes préférences (par exemple, si la satisfaction de [R2] seule est préférée à la satisfaction de [S6] seule) ? Comment prendre en compte les éventuelles synergies entre les préférences considérées ? Les principaux formalismes de l'art utilisés pour définir une relation de préférences $\succsim$ dans le cadre de la planification sont présentés dans la section 1.2.3.

1.2.3 Langage formel pour la planification avec préférences

Plusieurs formalismes pour raisonner sur les problèmes de planification avec préférences ont été proposés dans la littérature (voir [7] pour un état de l'art complet). Cette section présente en détails (syntaxe et sémantique) le PDDL3 [64] qui a enrichi les versions précédentes du langage par la prise en compte des préférences. Bien que le PDDL3 soit le langage le plus utilisé, les principales approches alternatives sont également présentées. Pour conclure, les différentes méthodes utilisées pour raisonner sur les préférences sont comparées les unes aux autres.

Syntaxe du langage PDDL3

Lors de la définition du PDDL3, la notation BNF du langage a été étendue pour permettre la représentation des préférences [63]. Ces modifications sont illustrées à l'aide de quelques extraits basés sur l'exemple introduit dans la section 1.2.2. La représentation complète de ce problème est disponible dans l'annexe B.3.

```
(:goal (and
  (possede_echantillon_sol N1 L4)
  (possede_echantillon_roche N1 L3)
  (possede_photographie N1 L1)

  (preference f1 (possede_echantillon_sol N1 L6))
  (preference f2 (possede_echantillon_roche N1 L2))
))
```

FIGURE 1.9 – Objectifs finaux et préférences finales en PDDL

```
(:constraints (and
  (sometime-before (possede_echantillon_roche N1 L3)
    (possede_echantillon_sol N1 L4))

  (preference s1 (sometime (position N1 L1)))
  (preference a1 (at-most-once (position N1 L6)))
))
```

FIGURE 1.10 – Objectifs et préférences de trajectoire en PDDL

Sémantique des préférences en PDDL3

Cette section enrichit la sémantique de la planification classique avec la notion de préférences. Les définitions 1.22 à 1.26 sont introduites dans ce but et sont construites par extension des définitions présentées dans la section 1.1.3.

Définition 1.22 - Instance d'un problème de planification [7]
Une instance d'un problème de planification avec préférences I_P est redéfinie par extension de la définition 1.14 en ajoutant les formules G' et SG ainsi qu'une relation d'ordre $\succsim$ à $Prob = (O, s_0, G', SG, \succsim)$ avec :

- G' une formule qui représente un ensemble d'objectifs étendus
- SG une formule qui représente un ensemble de préférences
- $\succsim$ une relation d'ordre construite à partir de SG

Dans le langage PDDL3, l'interprétation sémantique des préférences et des objectifs étendus s'appuie sur la notion de trajectoire qui s'interprète naturellement comme une succession de transitions d'états (cf. définition 1.11). Par ailleurs, elle repose sur des mécanismes analogues à ceux des logiques modales [24, 62] et s'inspire notamment de travaux de planification utilisant la logique temporelle [3].

Définition 1.23 - Valeur de vérité d'un objectif de trajectoire [64]
Étant donné un plan x et deux formules ϕ et ψ, la valeur de vérité des objectifs de trajectoire s'interprète par rapport à la trajectoire de x :

$$\langle s_0, \cdots, s_n \rangle \models \phi \quad \Leftrightarrow \quad s_n \models \phi$$

$$\langle s_0, \cdots, s_n \rangle \models (\texttt{always}\ \phi) \quad \Leftrightarrow \quad \forall i : 0 \leq i \leq n,\ s_i \models \phi$$

$$\langle s_0, \cdots, s_n \rangle \models (\texttt{sometime}\ \phi) \quad \Leftrightarrow \quad \exists i : 0 \leq i \leq n,\ s_i \models \phi$$

$$\langle s_0, \cdots, s_n \rangle \models (\texttt{at-most-once}\ \phi) \quad \Leftrightarrow \quad \forall i : 0 \leq i \leq n, \text{ si } s_i \models \phi, \text{ alors } \exists j : j \geq i, \forall k : i \leq k \leq j, s_k \models \phi \text{ et } \forall k : k > j,\ s_k \models \neg\phi$$

$$\langle s_0, \cdots, s_n \rangle \models (\texttt{sometime-before}\ \phi\ \psi) \quad \Leftrightarrow \quad \forall i : 0 \leq i \leq n, \text{ si } s_i \models \phi \text{ alors } \exists j : 0 \leq j < i,\ s_j \models \psi$$

Il existe également des opérateurs modaux temporels à l'image de `within` qui permet par exemple de préciser qu'une formule doit être vérifiée avant une date donnée. La sémantique des objectifs de trajectoire temporels est précisée dans [64] mais n'est pas reprise ici puisque ces derniers ne sont pas utilisés dans cette étude.

Les préférences étant des contraintes faibles (cf. section 1.2.1), elles sont toujours considérées comme vraies afin de ne pas impacter la validité des plans. En revanche, elles sont déterminantes pour définir l'utilité d'un plan.

Définition 1.24 - Valeur de vérité d'une préférence [64]
Etant donné un plan $\langle \hat{a}_1, \cdots, \hat{a}_n \rangle$ et une préférence $p \in SG$ définie à partir de la formule d'un objectif de trajectoire Φ, la valeur de vérité de p est toujours vraie, ce qui est dénoté par :

$$\langle s_0, \cdots, s_n \rangle \models (\texttt{preference}\ \Phi)$$

De plus, p est *satisfaite* si $\langle s_0, \cdots, s_n \rangle \models \Phi$ et *violée* dans le cas contraire.

Pour illustrer ces définitions, l'exemple introduit dans la section 1.2.2 est considéré. Soit $\langle s_0, \cdots, s_{10} \rangle$ la trajectoire correspondant au plan 2. De plus, soit ϕ_8 et ϕ_9 les deux prédicats décrits respectivement par `(position N1 L1)` et `(position N1 L6)`. Puisqu'il n'existe pas de $i \in [0, 10]$ tel que $s_i \models \phi_8$, la préférence `(sometime (position N1 L1))` est violée dans le plan 2. En revanche, $\forall i \in \{7; 8\}, s_i \models \phi_9$ et $\forall i \in [0; 6] \bigcup \{9; 10\}, s_i \models \neg\phi_9$ ce qui permet de déduire que la préférence `(at-most-once (position N1 L6))` est satisfaite par le plan 2.

Définition 1.25 - Validité d'un plan [64]
La notion de validité d'un plan $\langle \hat{a}_1, \cdots, \hat{a}_n \rangle$ est redéfinie par extension de la définition 1.13 en considérant que la formule des objectifs étendus G' doit être vraie par rapport à la trajectoire du plan à savoir $\langle s_0, \cdots, s_n \rangle \models G'$.

Définition 1.26 - Utilité d'un plan [7]
Soit X l'ensemble des plans valides de I_P et $U : X \to \mathbb{R}$ une fonction qui associe à tout plan $x \in X$ une valeur d'*utilité* sur la base des préférences de I_P. Si la fonction U doit être minimisée, alors un plan x_1 est préféré à x_2 noté $x_1 \succsim x_2$ si et seulement si $U(x_1) \leq U(x_2)$.

Domaine de la gestion de crise
En gestion de crise, les préférences PDDL3 *représentent des objectifs qu'il est souhaitable d'atteindre dans un plan solution sans que cela ne soit pour autant obligatoire. Par exemple, les décideurs peuvent préférer fournir un chauffage électrique à des victimes plutôt que de simples couvertures ou encore assurer une continuité de service complète si cela est possible (télécommunications, électricité...). Ces deux préférences peuvent être respectivement modélisées par une préférence finale et une préférence de trajectoire de type* `always`.

Autres formalismes de représentation des préférences

Le cas du PDDL3 ayant été traité, cette section s'intéresse à quelques formalismes alternatifs qui peuvent être employés dans le cadre de la planification avec préférences. Ces derniers ont tous pour objectif de définir une relation de préférence

permettant de distinguer les meilleurs plans parmi l'ensemble des plans solutions néanmoins leurs approches respectives diffèrent fortement. Ainsi, les *approches quantitatives* telles que la planification à satisfaction partielle (PSP) [130, 144], la planification basée sur la théorie de la décision [30] ou encore les approches basées sur le PDDL3 proposent d'associer une valeur numérique à chaque plan de sorte à pouvoir les comparer entre eux. Il devient alors facile d'estimer la qualité d'un plan puisqu'il suffit de calculer la valeur numérique qui lui est associée (cf. définition 1.26 par exemple). En revanche, dans le cas des *approches qualitatives*, la relation de préférence $x_1 \succsim x_2$ est définie uniquement à partir des propriétés des plans x_1 et x_2 sans qu'aucune valeur numérique ne leur soit associée. Les langages $\mathcal{PP}$ [131] et $\mathcal{LPP}$ [5, 23], les réseaux de préférences conditionnelles (CP-Nets) [29, 31] et les langages utilisant des bases de connaissances [56] sont des formalismes qualitatifs. Cette section ne s'intéresse qu'aux approches PSP, $\mathcal{LPP}$ et CP-Nets puisque ces dernières sont les alternatives au PDDL3 les plus couramment utilisées. Une description exhaustive de tous les langages utilisés dans le cadre de la planification avec préférences est disponible dans [7].

La planification à satisfaction partielle (PSP) [130, 144] étend le problème de la planification classique par l'introduction de deux fonctions U et C. Soit I_C une instance d'un problème de planification, la fonction $U : Atm_{I_C} \rightarrow \mathbb{R}^+$ associe à chaque atome de I_C une utilité positive. De même, la fonction $C : A \rightarrow \mathbb{R}^+$ définit un coût positif pour chaque action de I_C. Le bénéfice net d'un plan solution x est défini comme la différence entre la somme des utilités associées aux objectifs atteints dans x et la somme des coûts des actions de x. La relation de préférences sur l'ensemble des plans solutions d'un problème PSP est définie à partir du bénéfice net. Ainsi, un plan x_1 est préféré à un plan x_2 lorsque son bénéfice net est supérieur ou égal à celui de x_2.

Le langage $\mathcal{LPP}$ [5, 23] s'appuie sur le formalisme du *situation calculus*. Il permet d'exprimer des préférences finales ou de trajectoire avec une sémantique similaire à celle du PDDL3. Ces préférences élémentaires peuvent ensuite être composées au sein de formules de préférences atomiques. Ainsi, à partir de trois formules élémentaires ϕ_1, ϕ_2 et ϕ_3 et des niveaux de satisfaction $\{ excellent, satisfaisant, neutre \}$, il est possible de créer la formule de préférence atomique $\psi := \phi_1[excellent] \gg \phi_2[satisfaisant] \gg \phi_3[neutre]$. Le niveau de satisfaction de ψ dépend des valeurs

de vérités de ϕ_1, ϕ_2 et ϕ_3. Dans cet exemple, si ϕ_1 est satisfaite alors le niveau de satisfaction de ψ est *excellent*. En revanche, si ϕ_1 et ϕ_2 sont violées mais que ϕ_3 est satisfaite, le niveau de satisfaction associé à ψ est *neutre*. Finalement, plusieurs méthodes peuvent être employées pour agréger les formules de préférences atomiques. Certaines d'entre elles associent une valeur d'utilité aux plans ce qui confère alors une nature hybride (qualitative et quantitative) au langage.

Les réseaux de préférences conditionnels (CP-Nets) [29, 31] sont des représentations graphiques utilisées pour formaliser de façon compacte et intuitive les préférences des utilisateurs. Ils diffèrent beaucoup de l'approche PDDL3 en se focalisant sur des *préférences conditionnelles* dites *Ceteris Paribus*. De telles préférences permettent par exemple d'exprimer que *toute chose égale par ailleurs*, une formule ϕ_1 est préférée à ϕ_2 si ϕ_3 est vraie mais que ϕ_2 est préférée à ϕ_1 si ϕ_4 est vraie. En planification, elles peuvent être utilisées pour représenter des préférences entre des états objectifs. Ainsi, un plan x_1 est préféré à un plan x_2 si son état final s_n^1 est au moins aussi préféré que l'état final s_n^2 au sens du réseau de préférences conditionnelles considéré. Il convient de préciser que la relation d'ordre ainsi définie est partielle. En conséquence, il peut exister des plans qui ne peuvent être comparés avec cette relation de préférences.

Comparaison des formalismes de représentation des préférences

Cette section a pour but de comparer les formalismes de représentation de préférences précédemment listés. L'approche PDDL3/MAUT décrite en détails dans le chapitre 3 est également considérée afin de permettre au lecteur de la positionner dès à présent par rapport à l'état de l'art.

Les critères de comparaison retenus sont (i) la nature et le type de préférences qui peuvent être représentées, (ii) la méthode utilisée pour agréger ces dernières et (iii) la complexité de la construction du modèle de préférences correspondant. Une telle comparaison est nécessairement imparfaite puisque certains aspects des formalismes considérés sont par nature incomparables. Néanmoins, le prisme d'étude retenu permet de dresser un panorama général de l'emploi des préférences en planification.

En ce qui concerne la nature des préférences, les *représentations floues* qui permettent de préciser le degré de satisfaction d'une préférence sont à opposer aux *représentations binaires* qui ne peuvent exprimer que le fait qu'une préférence soit satisfaite ou non. Tous les formalismes mentionnés jusqu'à présent reposent sur la représentation binaire. En outre, plusieurs types de préférences peuvent être distinguées. Les *préférences finales*, *préférences de trajectoire* et *préférences numériques* représentent respectivement des choix du décideur qui portent sur l'état final à atteindre, les états intermédiaires du plan ou l'évolution d'une variable numérique du problème. A celles-ci viennent s'ajouter les *préférences conditionnelles* présentées dans la section précédente. De plus, le formalisme considéré peut permettre la *composition* de plusieurs préférences élémentaires afin de former des préférences plus riches à l'image des formules de préférences atomiques du langage $\mathcal{LPP}$.

Une fois que les préférences du décideur ont été spécifiées, il est nécessaire de les agréger afin de pouvoir comparer des plans entre eux. Agréger les préférences selon un *ordre lexicographique* est l'une des stratégies les plus intuitives pour les décideurs. Pour cela, les préférences sont classées par ordre de priorité $p_1, p_2, \cdots, p_n$. Le plan x_1 est alors préféré à x_2 si la valeur associée à la préférence p_1 dans x_1 est supérieure à celle de x_2. En cas d'égalité, la comparaison se poursuit avec p_2 et ce jusqu'à ce qu'un plan soit préféré à l'autre ou jusqu'à ce que toutes les préférences aient été considérées auquel cas les deux plans sont indifférenciés du point de vue du décideur. Bien que cette méthode soit très facile à mettre en œuvre, elle ne permet pas d'effectuer de compromis entre les différentes préférences du décideur. L'utilisation de méthodes basées sur une *pondération* entre les préférences du décideur permet d'adresser cette limite. La somme pondérée est généralement utilisée à cet effet mais des techniques plus génériques peuvent également être mises en œuvre ; ces dernières pouvant notamment rendre compte des éventuelles *interactions* qui existent entre les préférences du décideur.

Finalement, il est également crucial de considérer la complexité de la réalisation du modèle de préférences. Les préférences et les mécanismes d'agrégation utilisés peuvent être *simples* ou *complexes* à définir. Lorsque la construction est *manuelle*, l'utilisateur doit déterminer seul les différents paramètres du modèle de préférences utilisé. Au contraire, lorsque celle-ci est *outillée*, il peut s'appuyer sur un ensemble de méthodes scientifiques ou logicielles afin de réaliser son modèle de préférences.

	Préférence	**Agrégation**	**Modèle**
PDDL3	Binaire Finale, Trajectoire, *Numérique* Aucune composition	Pondération Pas interaction	Préférence : Simple *Agrégation : Complexe* Construction Manuelle
PDDL3 MAUT	**Floue** Finale, Trajectoire, **Numérique** **Composition**	Pondération **Interactions**	Préférence : Complexe Agrégation : Complexe **Construction Outillée**
PSP	Binaire Finale Aucune composition	Pondération Pas interaction	Préférence : Simple *Agrégation : Complexe* Construction Manuelle
$\mathcal{LPP}$	Binaire Finale, Trajectoire Composition	Ordre lexico. Pondération Pas interaction	*Préférence : Complexe* *Agrégation : Complexe* Construction Manuelle
CP-Nets	Binaire Finale, Cond. Aucune composition	Pondération Pas interaction	Préférence : Simple *Agrégation : Complexe* Construction Manuelle

TABLEAU 1.1 – Formalismes pour la planification avec préférences

Dans le tableau 1.1, les éléments de la colonne « Préférence » *en italique* indiquent que la fonctionnalité peut être considérée comme partiellement supportée. Par exemple en PDDL3, la plupart des modèles réalisés contiennent seulement une préférence numérique qui représente un coût global associé au plan. Bien que le langage permette d'utiliser un grand nombre de préférences numériques, la construction du modèle de préférences devient rapidement complexe dans de tels cas. Les éléments *en italique* de la colonne « Modèle » précisent les étapes de la construction du modèle qui peuvent être difficiles. Cette complexité est généralement liée au fait qu'il est demandé aux utilisateurs de fournir directement des paramètres numériques du modèle. Par exemple, lors de l'étape d'agrégation en PDDL3, les utilisateurs doivent renseigner les poids relatifs de leurs préférences afin de construire une somme pondérée. Il peut être très complexe pour les utilisateurs de fournir

ces informations de sorte que le modèle correspondant ait un sens opérationnel notamment quand le nombre de préférences devient important ou que ces dernières ne sont pas toutes du même type.

La ligne « PDDL3/MAUT » du tableau 1.1 correspond à une extension du langage PDDL à partir de mécanismes d'aide à la décision multicritère. Réalisée dans le cadre de cette étude et décrite en détails dans le chapitre 3, cette approche repose sur une représentation floue des préférences. Elle permet notamment de traiter le cas de plusieurs préférences numériques, de composer facilement des préférences élémentaires et de représenter des interactions entre préférences. Ce gain en pouvoir expressif augmente la complexité du modèle mais la construction de ce dernier peut être outillée de sorte à être simplifiée, voire devenir quasi-intuitive pour les utilisateurs.

1.2.4 Algorithmes de planification avec préférences

Cette section s'intéresse aux planificateurs et algorithmes utilisés pour résoudre les problèmes de planification avec préférences. La comparaison proposée s'appuie sur l'étude réalisée dans [7]. Les algorithmes retenus sont caractérisés par le formalisme de représentation des préférences et la stratégie de résolution qu'ils emploient. En outre, une attention particulière est portée aux algorithmes basés sur le langage PDDL3 puisque ces derniers peuvent être facilement comparés les uns aux autres à l'aide des problèmes de références utilisés lors des compétitions internationales de planification.

Comparaison des algorithmes de planification avec préférences

Plusieurs définitions sont introduites afin de caractériser les algorithmes de planification en fonction de leur stratégie de résolution des problèmes.

Définition 1.27 - Algorithme optimal [7]
Un algorithme A est optimal si pour tout problème $PP = (PT, \succsim)$, il retourne un plan optimal au sens de $\succsim$.

Beaucoup d'algorithmes ne sont pas conçus pour rechercher des plans de taille infinie et ne peuvent par conséquent pas être considérés comme optimaux au sens de la définition 1.27. Cependant, ils peuvent être capables de conduire à des solutions optimales pour des plans de taille donnée ce qui motive la définition 1.28. La taille d'un plan $\langle a_1, \cdots, a_n \rangle$ s'interprète comme le nombre d'actions présentes dans le plan à savoir n.

Définition 1.28 - Algorithme k-optimal [7]
Étant donné un entier positif k, un algorithme A est dit k-optimal si pour tout problème $PP = (PT, \succsim)$, il retourne un plan optimal au sens de $\succsim$ dont la taille est au plus k ; si un tel plan existe.

La recherche de solutions optimales pouvant être très complexe, une approche classique consiste à retourner une succession de plans suboptimaux de qualité croissante ; et ce jusqu'à ce qu'une solution optimale soit trouvée ou que les ressources à disposition (temps ou mémoire) soient épuisées.

Définition 1.29 - Algorithme itératif [7]
Un algorithme A est itératif s'il existe une famille non vide F de problèmes PP telle que pour tout problème $(PT, \succsim) \in F$, il retourne une séquence non vide et non singleton de plans $(x_1, x_2, \cdots)$ solutions de PP telle que :

$$\forall i, j \in \mathbb{N} \text{ tel que } 1 \leq i < j,\ x_j \succsim x_i \ \text{ et } \ x_i \not\succsim x_j.$$

Équipé de ces définitions, une comparaison des algorithmes de planification avec préférences peut être réalisée (cf. tableau 1.2). Le planificateur CHOPLAN décrit en détails dans le chapitre 3 est également considéré afin de permettre au lecteur de le comparer aux algorithmes de l'art.

Planificateur	Formalisme	Opti.	*k*-Opti.	Incrém.	Réf.
CHOPLAN	PDDL3	✓	-	✓	-
GAMER	PDDL3	✓	-	-	[51]
HPLAN-P	PDDL3	✓	✓	✓	[6]
HSP	PDDL3	✓	-	-	[78]
LPRPG-P	PDDL3	✓	-	✓	[39]
MIPS-BDD	PDDL3	✓	-	-	[48]
MIPS-XXL	PDDL3	✓	-	✓	[50]
SGPlan5	PDDL3	-	-	-	[83]
$Yochan^{PS}$	PDDL3	-	-	✓	[20]
BBOP-LP	PSP	✓	-	✓	[21]
$Sapa^{PS}$	PSP	-	-	✓	[144]
HPLAN-QP	$\mathcal{LPP}$	✓	✓	✓	[5]
PPLAN	$\mathcal{LPP}$	-	✓	-	[23]
PREFPLAN	CP-Nets	-	✓	-	[31]

TABLEAU 1.2 – Algorithmes pour la planification avec préférences [7]

Comparaison des planificateurs basés sur le langage PDDL3

Des compétitions sont organisées tous les deux ou trois ans par la communauté scientifique de la planification. Celles-ci permettent de comparer différents algorithmes de planification entre eux sur la base d'un ensemble de problèmes de références formalisés en PDDL. La 5$^{\text{ème}}$ compétition internationale de planification (IPC5) [65] qui s'est déroulée en 2006 a introduit la troisième version du PDDL ainsi que plusieurs domaines et problèmes pour la planification avec préférences. Pour résoudre les problèmes proposés lors de IPC5, les planificateurs devaient être capables de supporter tout ou partie de la spécification du PDDL3. En outre, des problèmes de planification avec préférences ont également été utilisés en 2008 au cours de la 6$^{\text{ème}}$ compétition internationale de planification [43]. Néanmoins, ces derniers ne sont pas aussi riches que ceux retenus pour IPC5 puisqu'ils se

limitent à l'utilisation de préférences finales. Enfin, par manque de participants, les problématiques de la planification avec préférences n'ont pas été adressées lors des $7^{\text{ème}}$ et $8^{\text{ème}}$ éditions de la compétition internationale de planification.

Le tableau 1.3 s'intéresse aux différents planificateurs de l'art basés sur le langage PDDL3. Il précise leur support des fonctionnalités relatives aux préférences du langage PDDL ainsi que leurs éventuelles participations aux compétitions internationales de planification. Il convient de remarquer qu'aucune distinction n'a été faite entre les planificateurs qui supportent l'intégralité des préférences de trajectoire et ceux qui ne sont capables de gérer que les préférences de trajectoire non temporelles à l'image de CHOPLAN par exemple.

Planificateur	**OT**	**PF**	**PT**	**PN**	**IPC**
HPLAN-P	-	✓	✓	-	$5^{\text{ème}}$
MIPS-BDD	-	✓	✓	-	$5^{\text{ème}}$
MIPS-XXL	✓	✓	✓	✓	$5/6^{\text{ème}}$
SGPlan5	✓	✓	✓	✓	$5^{\text{ème}}$
Yochan^{PS}	-	✓	-	-	$5^{\text{ème}}$
GAMER	-	✓	-	✓	$6^{\text{ème}}$
HSP	-	✓	-	✓	$6^{\text{ème}}$
LPRPG-P	✓	✓	✓	✓	-
CHOPLAN	✓	✓	✓	✓	-

TABLEAU 1.3 – Participation des planificateurs aux compétitions internationales (IPC) et support du langage PDDL3. « OT » signifie objectifs de trajectoire, « PF » signifie préférences finales, « PT » signifie préférences de trajectoire et « PN » signifie préférences numériques.

Conclusion

Ce chapitre a débuté par la description du modèle conceptuel de la planification qui formalise l'activité de la planification automatisée. Ce dernier propose un ensemble de huit hypothèses fondamentales qui, lorsqu'elles sont considérées simultanément, définissent la classe des problèmes dits de planification classique. De plus, le langage formel PDDL qui est utilisé pour représenter les problèmes de planification a été introduit. Une grande attention a été portée à la sémantique de ce dernier puisqu'elle permet de préciser les mécanismes et concepts employés pour résoudre les problèmes de planification. Par la suite, la notion de préférences a été présentée afin de décrire la problématique de la planification avec préférences qui est au cœur de cette étude. Les différents formalismes de représentation de préférences utilisés en planification ont été exposés en détails. Ces derniers sont fondamentaux puisqu'ils influent sur le type de problèmes qui peuvent être résolus ainsi que sur les méthodes et techniques employées pour les résoudre. Pour finir, les principaux algorithmes de planification avec préférences de l'art ont été présentés.

Le chapitre 2 introduit un formalisme de modélisation de préférences utilisé en aide à la décision multicritère et encore jamais employé dans le cadre de la planification. Celui-ci sera utilisé dans le chapitre 3 pour étendre le pouvoir expressif du langage PDDL3 ainsi que pour créer un algorithme original de résolution des problèmes de planification avec préférences (CHOPLAN).

Chapitre 2

Modélisation de préférences avec la Théorie de l'Utilité Multi-Attribut couplée à une intégrale de Choquet

Ce chapitre traite de la modélisation des préférences en aide à la décision multicritère. Il constitue en cela un état de l'art de ce domaine dont le périmètre a été volontairement restreint aux seules méthodes utilisées dans cette étude. La section 2.1 introduit les problèmes d'aide à la décision multicritère et présente une méthode de résolution en trois étapes : (*i*) la construction d'un ensemble d'attributs préférentiels, (*ii*) la réalisation d'un modèle de préférences et enfin (*iii*) la recherche des alternatives préférées.

Dans le cadre de cette étude, le choix des attributs préférentiels est laissé à la charge des décideurs. La construction d'un modèle de préférence basé sur la théorie de l'utilité multi-attribut (introduite en section 2.2) couplée à une intégrale de Choquet (introduite en section 2.3) est présentée dans la section 2.4. La question de la recherche d'alternatives préférées pour un problème de planification avec préférences basé sur la théorie de l'utilité multi-attribut couplée à une intégrale de Choquet est traitée dans le chapitre 3 de ce manuscrit.

2.1 Aide à la décision multicritère

2.1.1 De la problématique de l'aide à la décision multicritère

L'aide à la décision est un domaine d'étude de la recherche opérationnelle qui peut être défini comme « *l'activité de la personne qui, par l'utilisation de modèles mathématiques, aide les décideurs à obtenir des éléments de réponse au cours d'un processus de décision* » [123]. Les méthodes d'aide à la décision sont employées pour adresser de nombreux problèmes comme par exemple l'amélioration des architectures des systèmes complexes [110, 116] ou encore l'évaluation de plans de gestion de crise (cf. chapitre 4).

Dans la plupart des problèmes de décision, le décideur est amené à considérer plusieurs critères différents. Ces derniers pouvant éventuellement être contradictoires les uns avec les autres à l'image de la surface habitable et du coût dans le cas de l'acquisition d'un logement. La communauté de l'*aide à la décision multicritère* (MCDA) propose de nombreux modèles et outils pour représenter et résoudre de tels problèmes de décision [88, 118].

Les problèmes de décision multicritère sont généralement classés en quatre catégories : choix, tri, rangement et description [123]. Le problème du *choix* vise à déterminer l'ensemble (éventuellement singleton) des meilleures alternatives offertes au décideur. Le problème du *tri* cherche quant à lui à affecter à chaque alternative une catégorie de sorte à classer ces dernières selon une typologie préalablement déterminée. Dans le problème de *rangement*, le but est de définir une relation d'ordre entre les différentes alternatives de décision considérées. Finalement, la problématique de *description* consiste à fournir des informations relatives aux choix qui peuvent être effectués.

Quelques notations fréquemment utilisées en aide à la décision multicritère sont à présent présentées. L'*ensemble des alternatives* qui doivent être comparées les unes aux autres est dénoté X. Chaque alternative est caractérisée par p *attributs* numérotés selon l'*ensemble des attributs* $P = \{1, \cdots, p\}$. Pour chaque attribut k, un *espace de définition* $\Omega_k \subset \mathbb{R}$ est défini ainsi qu'une fonction $z_k : X \to \Omega_k$ qui associe à chaque alternative $x \in X$ sa valeur pour l'attribut k. De façon analogue,

la fonction $z : X \to \Omega$ associe à chaque alternative $x \in X$ un vecteur de l'espace des attributs $\Omega = \Omega_1 \times \cdots \times \Omega_p$. En conséquence, à chaque alternative $x \in X$ correspond un vecteur de $Y = \{z(x) = (z_1(x), \cdots, z_p(x)) \mid x \in X\}$. Cet ensemble $Y \subset \Omega$ est appelé l'*ensemble des alternatives dans l'espace des attributs*.

Le problème de la planification avec préférences (cf. chapitre 1) peut être considéré comme un problème de choix puisqu'il consiste à fournir au décideur au moins l'un des meilleurs plans au sens des préférences définies dans le problème. Par conséquent, seule la problématique du choix sera considérée dans la suite de cette étude. Dans ce contexte, le terme *solution* (ou encore plan) pourra être utilisé en lieu et place du terme alternative.

2.1.2 Résolution des problèmes d'aide à la décision multicritère

La résolution d'un problème de décision multicritère peut être décrite à travers les trois étapes suivantes : (*i*) la construction d'un ensemble d'attributs préférentiels, (*ii*) la réalisation d'un modèle de préférences et enfin (*iii*) la recherche des alternatives préférées. Chacune de ces étapes successives est présentée dans la suite de cette section.

Construction d'un ensemble d'attributs préférentiels

La première étape a pour vocation de déterminer les différents attributs préférentiels à utiliser ainsi que leurs espaces de définition. Il s'agit intrinsèquement d'une activité de modélisation et elle constitue ainsi le premier choix du décideur : l'élicitation des points de vues à considérer pour évaluer les différentes alternatives les unes par rapport aux autres.

L'ensemble des attributs préférentiels construit doit vérifier les propriétés d'exhaustivité, de cohésion et de non redondance [124]. L'ensemble des attributs est *exhaustif* s'il permet de comparer n'importe quel couple d'alternatives (x^a, x^b) ce qui correspond intuitivement à l'idée qu'aucun attribut n'a été oublié. Ainsi si pour tout attribut $k \in P$, $z_k(x^a)$ et $z_k(x^b)$ sont jugés identiques alors x^a et x^b sont indifférenciés par rapport à l'ensemble des attributs P.

La propriété de *cohésion* traduit l'idée qu'aucun attribut n'est inutile. Il existe pour chaque attribut $k \in P$ au moins un couple d'alternative (x^a, x^b) tel que x^a soit préféré à x^b avec $z_k(x^a)$ étant préféré à $z_k(x^b)$ et $z_j(x^a)$ étant jugé identique à $z_j(x^b)$ pour tout $j \in P \backslash \{k\}$.
L'ensemble des attributs est *non redondant* s'il ne contient aucun attribut identique. Par conséquent, la suppression d'un attribut entraîne nécessairement la violation d'une des deux propriétés précédentes.

Par ailleurs, il est également possible de hiérarchiser les attributs retenus en formant des sous-ensembles d'attributs alors utilisés pour définir un attribut unique. Par exemple, dans le cas de la comparaison d'étudiants, un décideur peut utiliser comme attributs des notes de mathématique, de physique et de langue (l'espace de définition pouvant être l'intervalle $[0, 20]$). En regroupant les notes de mathématique et de physique, le décideur peut obtenir une note de science créant ainsi une hiérarchie au sein des attributs.

Réalisation d'un modèle de préférences

Une fois l'ensemble des attributs déterminé, il devient possible de réaliser un modèle des préférences du décideur. Un tel modèle est une relation d'ordre permettant de comparer automatiquement des alternatives entre elles. Afin de présenter la construction de ce modèle, la notion de *pré-ordre* est introduite.

Définition 2.1 - Pré-ordre
La relation binaire $\succsim \subset X \times X$ est un pré-ordre sur l'ensemble X ssi :

- $\forall x \in X,\ x \succsim x$ (*réflexivité*)
- $\forall x^a, x^b, x^c \in X,\ x^a \succsim x^b$ et $x^b \succsim x^c \Rightarrow x^a \succsim x^c$ (*transitivité*)

A partir d'un pré-ordre $\succsim$, les relations $\succ$ et $\sim$ sont définies telles que :

- $\forall x^a, x^b \in X,\ x^a \succ x^b \Leftrightarrow x^a \succsim x^b$ et $\neg(x^b \succsim x^a)$
- $\forall x^a, x^b \in X,\ x^a \sim x^b \Leftrightarrow x^a \succsim x^b$ et $x^b \succsim x^a$

Un pré-ordre est dit complet si pour toute paire d'éléments $x^a, x^b \in X$, la relation $\succsim$ est définie ($x^a \succsim x^b$ ou $x^b \succsim x^a$).

D'après des études réalisées en psychologie [129], il est plus facile pour les décideurs d'exprimer des préférences ordinales (comparaison d'alternatives entres elles) que des préférences cardinales (évaluation d'alternatives selon une échelle de satisfaction). La relation $\succsim_D$ (respectivement $\succ_D$ et $\sim_D$) représente les préférences ordinales du décideur. Elle est définie sur l'ensemble des alternatives X et son interprétation est la suivante :

- $x^a \succsim_D x^b$ signifie que x^a est au moins aussi préférée que x^b ;
- $x^a \succ_D x^b$ signifie que x^a est strictement préférée à x^b ;
- $x^a \sim_D x^b$ signifie que le décideur n'a pas de préférence entre x^a et x^b.

Dans un souci de simplification, la relation $\succsim_D$ est supposée définie sur Ω de sorte que $x^a \succsim_D x^b \Leftrightarrow y^a \succsim_D y^b$ avec $y^a, y^b \in Y$ où $Y = \{z(x) \mid x \in X\}$.

Le but de l'étape de réalisation du modèle de préférences est de construire un pré-ordre complet $\succsim$ qui respecte les préférences du décideur c'est à dire tel que : $\forall x^a, x^b \in X,\ x^a \succsim_D x^b \Rightarrow x^a \succsim x^b$. Au vue de la proximité entre ces relations, pourquoi est-il nécessaire de construire un tel modèle de préférences ? Il convient de remarquer que $\succsim$ est un modèle de $\succsim_D$ et constitue à ce titre une abstraction de cette relation de préférence. Ceci implique qu'il est possible de comparer deux alternatives x^a et x^b automatiquement au sens de $\succsim$ alors que la présence du décideur est requise pour les comparer au sens de $\succsim_D$. Si l'ensemble des alternatives est grand, il devient impossible de demander au décideur de comparer toutes les alternatives manuellement et le recours au modèle de préférences est indispensable. Tout l'art de cette étape consiste donc à construire un pré-ordre générique sur X à partir d'informations préférentielles portant sur un petit nombre d'alternatives.

Les techniques de réalisation de modèles de préférences proposées dans la littérature [57] peuvent être classées en deux catégories : les méthodes de surclassement et les méthodes de critère unique de synthèse. Les *méthodes de surclassement* telles que les méthodes ELECTRE [58, 59] et PROMETHEE [146] suivent l'approche dite « *Comparer puis Agréger* ». Elles consistent à comparer les attributs des alternatives deux à deux puis à agréger les comparaisons ainsi obtenues afin de déterminer si une alternative est préférée à une autre. Les *méthodes de critère unique de synthèse* à l'image de AHP [141] et de la Théorie de l'Utilité Multi-Attribut (MAUT) [45, 147] reposent quant à elles sur une approche de type « *Agréger puis*

Comparer ». Ces méthodes agrègent tous les attributs d'une alternative entre eux afin d'obtenir un critère unique de synthèse. Ce dernier définit alors un pré-ordre complet sur X qui permet de comparer toutes les alternatives entre elles.

Les deux approches présentent des avantages et des inconvénients. Les méthodes de surclassement sont particulièrement indiquées quand les différents attributs ne sont pas commensurables puisque les comparaisons effectuées portent toujours sur un même critère. En revanche, elles sont difficilement utilisables lorsque le nombre d'alternatives considérées est grand puisqu'il est nécessaire de comparer toutes les alternatives deux à deux pour identifier la meilleure d'entre elles. Les méthodes de critère unique de synthèse ne sont pas soumises à cette problématique puisque l'évaluation d'une alternative ne dépend pas des évaluations des autres alternatives. Ainsi, dans le cadre d'une problématique de choix, les méthodes de critère unique de synthèse permettent rapidement de savoir si une alternative est meilleure qu'une autre puisqu'il suffit de comparer cette dernière à la meilleure alternative connue.

La problématique de la planification avec préférences au cœur de cette étude étant un problème de choix, les méthodes de surclassement ne sont plus abordées dans la suite de ce document. De plus, seule la Théorie de l'Utilité Multi-Attribut MAUT est considérée dans le cadre de cette étude. Cette dernière a été retenue en raison de son grand pouvoir expressif (cf. sections 2.2 et 2.3).

Recherche des alternatives préférées

Equipé d'un ensemble d'attributs préférentiels défini par le décideur, il est possible de construire un modèle de préférences $\succsim$. Il convient à présent de s'intéresser à l'exploitation de ce modèle dans le cadre de la recherche des meilleures solutions d'un problème de choix donné.

La procédure qui permet d'obtenir des solutions du problème considéré est dénotée A. Bien que cette procédure puisse être quelconque (rien n'interdit qu'elle soit manuelle par exemple), il s'agit généralement d'un algorithme d'optimisation multi-objectif. En outre, la méthode pour comparer des alternatives au sens de $\succsim$ est notée MCDA. Dans le cadre de cette étude, A est un algorithme de résolution d'un problème de planification avec préférences (cf. section 1.2.2). Les alternatives sont donc les plans retournées par A.

La procédure A et la méthode MCDA peuvent être articulées de différentes manières (cf. figure 2.1). Dans une approche *a posteriori* (cf. [35] par exemple), la procédure A est mise en œuvre pour obtenir un ensemble X de solutions du problème. La méthode MCDA est ensuite utilisée sur X afin de déterminer l'ensemble des solutions préférées $X^{\succsim}$. Dans une approche *a priori* (voir [97, 98] par exemple), la méthode MCDA retourne un modèle de préférences qui devient alors un paramètre de la procédure A. Cette dernière peut ensuite être invoquée pour obtenir un ensemble de solutions préférées $X^{\succsim}$. Il convient également de mentionner les approches *interactives* [145] qui permettent de trouver un ensemble de solutions préférées sans construire entièrement le modèle de préférences. Ces dernières ne sont pas considérées car elles nécessitent une forte implication du décideur lors de la recherche des solutions ce qui ne correspond pas aux cas applicatifs envisagés pour cette étude.

L'approche a posteriori est généralement employée lorsque les informations préférentielles du décideur ne sont pas disponibles au moment de la résolution du problème. Dans le cas contraire, on privilégie l'approche a priori puisqu'elle permet de construire un ensemble des solutions préférées $X^{\succsim}$ sans avoir à générer d'ensemble de solutions X au préalable. En effet, cette étape peut s'avérer extrêmement coûteuse notamment dans le cas où l'ensemble X obtenu contient toutes les solutions du problème considéré. Le problème de la planification avec préférences étant fortement combinatoire, il n'est pas envisageable de construire l'ensemble de ses solutions. Par conséquent, l'approche a priori est le paradigme retenu dans le cadre de cette étude.

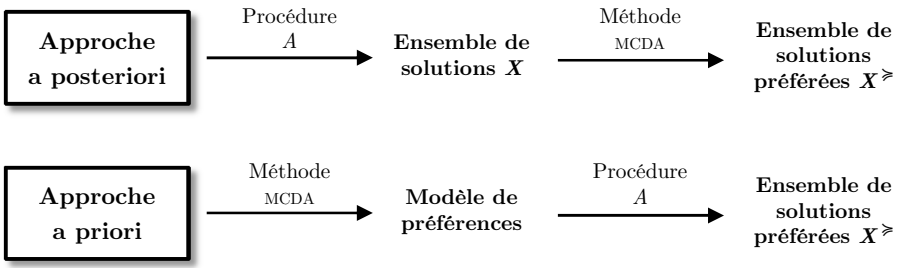


FIGURE 2.1 – Couplage d'une procédure A et d'une méthode MCDA [60]

Domaine de la gestion de crise
Proposer des plans d'action aux décideurs de la gestion de crise constitue une activité d'aide à la décision. En effet, il faut pour cela choisir un ou plusieurs plans (le ou les meilleurs par rapport à un modèle de préférences) parmi l'ensemble des plans pouvant adresser le problème.

Lors de la réponse à une situation de crise, les décideurs mobilisés sont fortement sollicités et peu disponibles. Une approche a priori est donc parfaitement indiquée puisqu'elle n'est que peu chronophage pour ces derniers ; la modélisation de leurs préférences pouvant avoir été réalisée au moins partiellement en amont de la crise.

2.2 Théorie de l'Utilité Multi-Attribut

Cette section s'intéresse uniquement à la question de la modélisation des préférences du décideur dans le cadre de la théorie de l'utilité multi-attribut. Les attributs préférentiels et leurs espaces de définition (tout deux requis pour cette étape) sont supposés préalablement déterminés.

2.2.1 Modèle MAUT

La théorie MAUT appartient à la classe des méthodes de critère unique de synthèse. Ainsi, elle associe à chaque alternative $x \in X$ une valeur numérique appelée *utilité* qui traduit le degré de qualité de x au sens des préférences du décideur et s'exprime sur une *échelle de satisfaction commune* $\xi \subset \mathbb{R}$.

Définition 2.2 - Modèle MAUT [45]
Soit X un ensemble d'alternatives, Ω un espace d'attributs et Y l'ensemble des alternatives X dans l'espace des attributs Ω défini par $Y = \{z(x) \mid x \in X\}$. Le modèle MAUT définit un pré-ordre complet $\succsim$ sur X par :

$$\begin{cases} x^a \succsim x^b \Leftrightarrow U(y^a) \geq U(y^b) \\ U(y) = \psi(u_1(y_1), \cdots, u_p(y_p)) \end{cases}$$

Avec $U : \Omega \rightarrow \mathbb{R}$ une fonction d'utilité MAUT construite à partir de :

- $u_k : \Omega_k \rightarrow \xi$ des fonctions d'utilité partielles ;
- $\psi : \xi^p \rightarrow \mathbb{R}$ une fonction d'agrégation.

Un *critère* k est la donnée d'un attribut $k \in P$ et d'une *fonction d'utilité partielle* u_k. Par abus de langage, P sera utilisé indifféremment pour désigner l'ensemble des attributs et celui des critères. Les critères peuvent être rendus commensurables puisque tous définis sur l'échelle de satisfaction commune ξ.

2.2.2 Fonctions d'utilité partielles

Les fonctions d'utilité partielles $u_k : \Omega_k \rightarrow \xi$ sont des fonctions à valeur dans l'échelle de satisfaction commune. Afin de les caractériser, il faut donc préciser la nature de l'ensemble $\xi \subset \mathbb{R}$.

Plusieurs approches peuvent être utilisées pour définir des degrés de satisfaction sur ξ. Une échelle de satisfaction est dite unipolaire [93, 94] s'il est possible de définir pour tout $k \in P$, deux éléments $\mathbf{0}_k$ et $\mathbf{1}_k$ de Ω_k tels que $\mathbf{0}_k$ (respectivement $\mathbf{1}_k$) est considéré comme totalement insatisfaisant (respectivement parfaitement satisfaisant) par le décideur pour le critère k.

De même, une échelle de satisfaction est dite bipolaire [72] s'il est possible de définir pour tout $k \in P$, un élément neutre $\mathbf{N}_k$ tel que les éléments de Ω_k préférés à (respectivement moins préférés que) $\mathbf{N}_k$ sont considérés « bons » (respectivement « mauvais ») par le décideur.

La nature de l'échelle de satisfaction choisie traduit un choix de modélisation. Dans ces travaux, l'échelle de satisfaction unipolaire $\xi = [0, 1]$ a été retenue. Cette échelle est l'une des plus utilisée notamment parce qu'elle est relativement intuitive pour les décideurs. De plus, par convention, l'utilité associée à $\mathbf{0}_k$ et $\mathbf{1}_k$ est définie de sorte que $u_k(\mathbf{0}_k) = 0$ et $u_k(\mathbf{1}_k) = 1$. Il convient néanmoins de préciser que tous les résultats présentés dans ce chapitre sont généralisables au cas de l'utilisation d'une échelle bipolaire [73].

A présent que la nature de l'échelle de satisfaction ξ est connue, il reste à préciser l'expression des fonctions u_k. Dans une première approche, il est possible d'utiliser une fonction d'utilité partielle linéaire définie par $u_k(y_k) = \lambda_k \times y_k,\ \forall y_k \in \Omega_k$. Ce modèle n'est malheureusement pas pleinement satisfaisant puisque la satisfaction du décideur n'évolue pas nécessairement de façon linéaire. Ainsi, dans le cadre de l'évaluation d'étudiants, un décideur peut largement préférer une note de 16 à une note de 12 mais ne préférer que modérément une note de 20 à une note de 16.

Pour outrepasser cette limite, il est possible de considérer des fonctions d'utilité partielles continues et *linéaires par morceaux*. Ces fonctions d'utilité possèdent un grand pouvoir expressif puisqu'elles permettent d'approximer n'importe quelle autre fonction arithmétique.

Définition 2.3 - Fonction d'utilité linéaire par morceaux [85]
Soit $u_k : \Omega_k \to \xi$ la fonction d'utilité partielle de l'attribut $k \in P$ et $\xi = [0, 1]$ une échelle de satisfaction. L'ensemble Ω_k est divisé en n intervalles dont les bornes $z_k^0 = 0, \cdots, z_k^n = 1$ sont définies telles que : $\forall i \in \{0, \cdots, n-1\},\ z_k^i < z_k^{i+1}$.

L'utilité d'une alternative $x \in X$ sur le critère k est définie par l'interpolation linéaire de $z_k(x) \in [z_k^i,\ z_k^{i+1}]$ avec $i \in \{1, \cdots, n\}$:

$$u_k(y_k) = u_k(z_k^i) + \frac{u_k(z_k^{i+1}) - u_k(z_k^i)}{z_k^{i+1} - z_k^i} \times (z_k(x) - z_k^i)$$

Dans la définition 2.3, l'utilité des différentes bornes $u_k(z_k^1), \cdots, u_k(z_k^n)$ est définie préalablement. Ainsi, il est possible de traiter le cas où Ω_k est un ensemble discret en considérant qu'à chaque valeur $\omega \in \Omega_k$ correspond une borne z_k^i dont la valeur est connue (l'interpolation linéaire qui n'est pas définie quand Ω_k est discret n'est alors plus nécessaire).

La construction des fonctions d'utilité (et notamment le calcul de l'utilité des bornes) peut être réalisée à l'aide de la méthode présentée en section 2.4.

2.2.3 Fonction d'agrégation

Dans le cadre MAUT, le rôle de la fonction d'agrégation $\psi : \xi^p \rightarrow \xi$ est de déterminer l'utilité d'une alternative sur la base de la valeur des critères retournés par les fonctions d'utilité partielles u_k. La somme pondérée est couramment utilisée comme fonction d'agrégation car elle possède l'avantage d'être facile à mettre en œuvre.

Définition 2.4 - Somme pondérée
Soit un vecteur $a \in \mathbb{R}^p$ et un vecteur de poids $w \in [0,1]^p$ tel que $\sum_{k=1}^{p} w_k = 1$ avec $w_k \geq 0$. La somme pondérée $S_w : \mathbb{R}^p \rightarrow \mathbb{R}$ est définie par :

$$S_w(a_1, \cdots, a_p) = \sum_{k=1}^{p} w_k \, a_k$$

Il est aussi possible d'utiliser d'autres opérateurs d'agrégation usuels tels que le minimum, le maximum ou encore la somme pondérée ordonnée (OWA) [148].

2.2.4 Exemple illustratif

Pour illustrer les propos de cette section, l'exemple (inspiré de [71]) d'un directeur de faculté souhaitant évaluer ses étudiants est utilisé. Dans ce contexte, les alternatives du problème de décision sont les étudiants et les attributs préférentiels du décideur sont les notes des étudiants en mathématique (M), physique (P) et langue (L). Les attributs préférentiels (M) et (P) sont exprimées sur l'espace de définition $\Omega_{MP} = [0, 20]$ et l'attribut (L) est exprimé sur l'espace de définition discret $\Omega_L = \{A, B, C, D, E\}$. Le décideur souhaite comparer trois étudiants E_a, E_b et E_c dont les notes sont renseignées dans le tableau suivant.

	Mathématique (M)	Physique (P)	Langue (L)
Etudiant E_a	16	16	B
Etudiant E_b	18	18	C
Etudiant E_c	16	15	B

TABLEAU 2.1 – Exemple : Notes des étudiants

Un modèle de préférence MAUT est à présent réalisé pour préciser les notions introduites précédemment. Le directeur utilise la même fonction d'utilité partielle $u_{MP} : [0, 20] \to [0, 1]$ pour les critères (M) et (P). Il est parfaitement satisfait lorsque la note d'un étudiant vaut $\mathbf{1}_{MP} = 20$. En revanche, si la note d'un étudiant est inférieure ou égale à $\mathbf{0}_{MP} = 8$, le directeur est totalement insatisfait. Par définition, $u_{MP}(8) = 0$ et $u_{MP}(20) = 1$. En outre, il préfère fortement une note de 16 à une note de 12 alors qu'il ne préfère que modérément une note de 12 (respectivement 20) à une note de 8 (respectivement 16). En ce qui concerne le critère (L), le directeur est parfaitement satisfait si l'étudiant a obtenu la note $\mathbf{1}_L = A$ et totalement insatisfait s'il a obtenu la note $\mathbf{0}_L = E$. Par définition, $u_L(E) = 0$ et $u_L(A) = 1$. En outre, le niveau de satisfaction du directeur varie linéairement entre A et E. Les fonctions d'utilité u_{MP} et u_L (cf. figure 2.2) modélisent les préférences du décideur et peuvent être construites à l'aide de la méthode présentée dans la section 2.4.1.

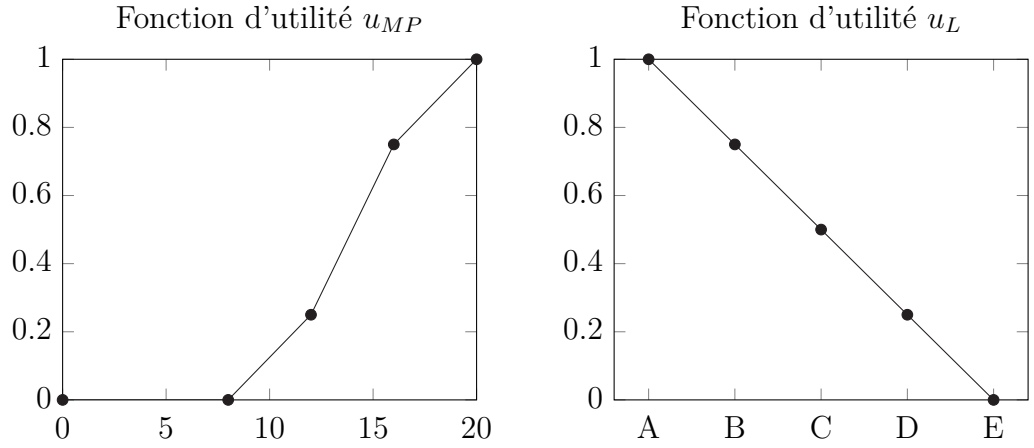


FIGURE 2.2 – Exemple : Fonctions d'utilité partielles u_{MP} et u_L

En utilisant u_{MP} et u_L, la valeur des différents critères peut être calculée :

	Mathématique (M)	Physique (P)	Langue (L)
Etudiant E_a	0.75	0.75	0.75
Etudiant E_b	0.875	0.875	0.5
Etudiant E_c	0.75	0.625	0.75

TABLEAU 2.2 – Exemple : Valeurs d'utilités partielles

Pour compléter le modèle MAUT, il faut à présent définir une fonction d'agrégation qui va permettre de déterminer l'utilité associée à chaque étudiant. Le directeur utilise pour cela une somme pondérée S_w dont le vecteur de poids est $w = (w_M, w_P, w_L) = (0.3, 0.3, 0.4)$. Ce choix traduit la volonté du directeur d'accorder la même importance aux mathématiques et à la physique dans son évaluation. De plus, le poids accordé aux langues (40% de la pondération) est élevé car le décideur préfère les étudiants au profil équilibré plutôt que les étudiants performants uniquement en sciences. L'utilisation de S_w fournit les résultats suivants :

	Etudiant E_a	Etudiant E_b	Etudiant E_c
Utilité	0.75	0.725	0.7125

TABLEAU 2.3 – Exemple : Utilités calculées à l'aide d'une somme pondérée

Ainsi, au sens de ce modèle MAUT construit à partir de u_{MP}, u_L et S_w, les préférences du directeur au regard de ses étudiants sont : $E_a \succsim E_b \succsim E_c$.

2.3 Intégrale de Choquet

La section 2.1 a introduit les trois étapes de la résolution des problèmes de décision. La modélisation des préférences du décideur (deuxième étape de la résolution) peut être réalisée en utilisant la théorie MAUT comme présenté dans la section 2.2.1. Cette section complète la section 2.2.1 en présentant une fonction d'agrégation pouvant être utilisée dans le cadre MAUT.

2.3.1 Limites de la somme pondérée

La somme pondérée bien que couramment utilisée est un opérateur d'agrégation aux possibilités limitées qui ne peut représenter certaines préférences des décideurs. Les limites de la somme pondérée peuvent être illustrées (exemple emprunté à [73]) à l'aide de deux critères dont les fonctions d'utilités u_1 et u_2 sont connues et de trois alternatives x^a, x^b et x^c à comparer telles que :

$$u_1(x_1^a) = 0.4 \qquad u_1(x_1^b) = 0 \qquad u_1(x_1^c) = 1$$
$$u_2(x_2^a) = 0.4 \qquad u_2(x_2^b) = 1 \qquad u_2(x_2^c) = 0$$

Si le décideur souhaite qu'aucune alternative ne comporte de critères non satisfaits ($\forall k \in P$, $u_k(x) \neq 0$), alors ses préférences sont $x^a \succ x^b \sim x^c$. Soit une somme pondérée S_w caractérisée par le vecteur de poids (w_1, w_2), les préférences du décideur induisent les deux relations suivantes :

$$\begin{aligned} x^b \sim x^c &\Leftrightarrow w_1 = w_2 \\ x^a \succ x^b &\Leftrightarrow 0.4\,(w_1 + w_2) > w_2 \end{aligned}$$

Ces deux relations considérées conjointement démontrent que la somme pondérée ne peut représenter les préférences de ce décideur puisqu'elles imposent $0.8\,w_2 > w_2$ ce qui est impossible.

2.3.2 Modèle basé sur l'intégrale de Choquet

Afin d'aboutir à un modèle plus général et plus expressif, la notion de *fonction de capacité* [36] (ou *capacité* ou encore mesure floue [132]) est introduite. Cette dernière généralise l'idée de poids de la somme pondérée en définissant un poids non seulement pour chaque critère mais également pour chaque ensemble de critères qu'il est possible de former à partir de P. La fonction de capacité μ est donc définie sur l'ensemble des parties de P noté $\mathfrak{P}(P)$.

Définition 2.5 - Fonction de capacité [73]
Soit P un ensemble de critères, la fonction d'ensemble $\mu : \mathfrak{P}(P) \rightarrow [0, 1]$ est une fonction de capacité si :

- $\mu(\emptyset) = 0$ et $\mu(P) = 1$
- $\forall A, B \subseteq P,\ A \subseteq B \ \Rightarrow\ \mu(A) \leq \mu(B)$

Au regard de ces deux propriétés, la fonction de capacité est dite normalisée et monotone ce qui s'interprète naturellement. En effet, le poids de l'ensemble vide est nul alors que le poids de l'ensemble des critères P est maximal. De plus, si l'ensemble de critères A est un sous ensemble de B alors son poids est nécessairement inférieur ou égal à celui de B. Par mesure de simplicité, le poids du critère k sera noté $\mu(k)$ plutôt que $\mu(\{k\})$. Pour deux critères 1 et 2, la propriété de monotonicité impose donc : $\mu(1) \leq \mu(\{1, 2\})$.

Equipé de la notion de capacité, il est à présent possible de définir l'*intégrale de Choquet* [36]. Dans le cadre de cette étude, cet outil mathématique est utilisé comme une fonction d'agrégation.

Définition 2.6 - Intégrale de Choquet [73]
Soit un vecteur $a \in \mathbb{R}_+^p$ et μ une fonction de capacité définie sur P. L'intégrale de Choquet $C_\mu : \mathbb{R}_+^p \to \mathbb{R}_+$ est définie par :

$$C_\mu(a_1, \cdots, a_p) = \sum_{k=1}^{p} \mu(\{\sigma(k), \cdots, \sigma(p)\}) \times [\, a_{\sigma(k)} - a_{\sigma(k-1)} \,]$$

Avec σ une permutation sur P telle que $a_{\sigma(1)} \leq \cdots \leq a_{\sigma(p)}$ et $a_{\sigma(0)} = 0$.

Si la notion de capacité s'interprète naturellement comme une généralisation de l'idée de poids d'une somme pondérée, l'interprétation de l'intégrale de Choquet n'est pas évidente de prime abord. Ainsi, le lecteur curieux peut légitimement se demander pourquoi l'intégrale de Choquet fait-elle l'objet d'une attention particulière au sein de la communauté MCDA ?

L'ensemble $X_A = \{\, (\mathbf{1}_A, \mathbf{0}_{-A}) \mid A \subseteq P \,\}$ est appelé ensemble des *alternatives binaires*. Les alternatives binaires correspondent à des alternatives qui seraient parfaitement satisfaisantes sur tous les critères $k \in A$ et totalement insatisfaisantes sur les autres. Etant donné que $u_k(\mathbf{1}_k) = 1$ et $u_k(\mathbf{0}_k) = 0$, l'utilité d'une alternative binaire se réduit à $\psi(1_A, 0_{-A})$. En outre, la définition de μ permet d'écrire que $\psi(1_A, 0_{-A}) = \mu(A)$. De plus, puisque μ est définie sur $\mathfrak{P}(P)$, elle précise l'utilité de l'ensemble des alternatives binaires de P et détermine ainsi ψ sur tous les sommets de l'hypercube $[0, 1]^p$. Par conséquent, la recherche des valeurs de ψ peut être vue comme un problème d'interpolation entre les sommets de l'hypercube $[0, 1]^p$. La solution de ce problème la plus simple (interpolation linéaire utilisant le moins de sommets possible pour chaque point) est unique et n'est autre que l'intégrale de Choquet [100, 103, 127].

L'intégrale de Choquet généralise les principales fonctions d'agrégation usuelles telles que le min, le max, la somme pondérée et la somme pondérée ordonnée OWA [68]. De plus, elle constitue une fonction d'agrégation très puissante puisqu'elle permet de modéliser la pondération, la complémentarité et la substituabilité de critères (ainsi que la notion de véto [70] par conséquence).

L'exemple illustrant les limites de la somme pondérée [73] qui a été présenté dans la section précédente peut facilement être résolu à l'aide de l'intégrale de Choquet. Les préférences du décideurs imposent cette fois :

$$\begin{aligned} x^b \sim x^c &\Leftrightarrow \mu(1) = \mu(2) \\ x^a \succ x^b &\Leftrightarrow 0.4\ \mu(\{1,2\}) > \mu(2) \end{aligned}$$

Par définition, $\mu(\{1,2\}) = 1$ car $P = \{1,2\}$. En prenant $\mu(1) = \mu(2) = 0.3$, il est donc possible de construire un modèle des préférences du décideur.

2.3.3 Interprétation du modèle

Le modèle de préférences basé sur une capacité et l'intégrale de Choquet est très riche. Malheureusement, ce grand pouvoir expressif a été obtenu au détriment de la simplicité du modèle. En effet, $2^p - 2$ paramètres sont à présent nécessaires pour déterminer la fonction de capacité entièrement (*i.e.* les différentes valeurs de μ sur $\mathfrak{P}(P)$ à l'exception des ensembles $\emptyset$ et P).

Afin de faciliter l'interprétation de μ (et donc *in fine* son élicitation), les notions de *valeur de Shapley* (ou *indice d'importance*) [126] et d'*indice d'interaction entre critères* [70] sont introduites. Ces deux notions seront particulièrement utiles lorsque le modèle sera simplifié dans la section 2.3.4.

La valeur de Shapley $\phi(k)$ exprime l'importance globale du critère k pour la capacité μ. Il convient de ne pas confondre $\phi(k)$ qui représente le poids total du critère k avec $\mu(k)$ qui traduit l'importance du critère k seul.

Définition 2.7 - Valeur de Shapley [126]
Soit μ une capacité, la valeur de Shapley du critère $k \in P$ est définie par :

$$\phi(k) = \sum_{B \subseteq P \setminus k} \frac{(|P| - |B| - 1)!\ |B|!}{|P|!} \times \Big[\mu(B \cup k) - \mu(B)\Big]$$

L'*indice d'interaction entre critères* généralise la valeur de Shapley (il suffit de remarquer que $I(\{k\}) = \phi(k)$) et apporte les informations complémentaires qui sont requises pour interpréter complètement le modèle. Il exprime le fait que l'importance d'un ensemble de critères $A \subseteq P$ ne se réduit pas à la somme des importances des critères $k \in A$.

Définition 2.8 - Indice d'interaction entre critères [70]
Soit μ une capacité, l'indice d'interaction de l'ensemble $A \subseteq P$ est défini par :

$$I(A) = \sum_{B \subseteq P \setminus A} \frac{(|P| - |B| - |A|)!\ |B|!}{(|P| - |A| + 1)!} \times \sum_{K \subseteq A} (-1)^{|A \setminus K|}\ \mu(B \cup K)$$

L'exemple proposé dans [74] illustre le phénomène d'interaction entre critères dans le cas $p = 2$. Soit deux critères ayant le même indice d'importance et quatre alternatives x^a, x^b, x^c, x^d (cf. figure 2.3) telles que :

$$y^a = (\mathbf{0}_1, \mathbf{0}_2) \qquad y^b = (\mathbf{0}_1, \mathbf{1}_2) \qquad y^c = (\mathbf{1}_1, \mathbf{0}_2) \qquad y^d = (\mathbf{1}_1, \mathbf{1}_2)$$

Il apparait clairement que $x^d \succ x^a$ mais le cas des alternatives x^b et x^c est plus complexe. Si le décideur considère que $x^a \sim x^b \sim x^c$ (cf. figure 2.3(a)), cela signifie que les deux critères doivent être satisfaits conjointement pour qu'une alternative soit jugée satisfaisante. Les critères sont alors dit *complémentaires* et leur indice d'interaction I est positif.

Si le décideur considère que $x^b \sim x^c \sim x^d$ (cf. figure 2.3(b)), alors il n'est nécessaire de satisfaire qu'un seul des deux critères pour qu'une alternative soit jugée satisfaisante. Les critères sont alors dit *substituables* et leur indice d'interaction I est négatif.

Dans le cas représenté sur la figure 2.3(c), chaque critère apporte sa propre contribution à la satisfaction générale de l'alternative. Les critères sont alors dits *indépendants* (comme dans le cas de la somme pondérée) et leur indice d'interaction I est nul.

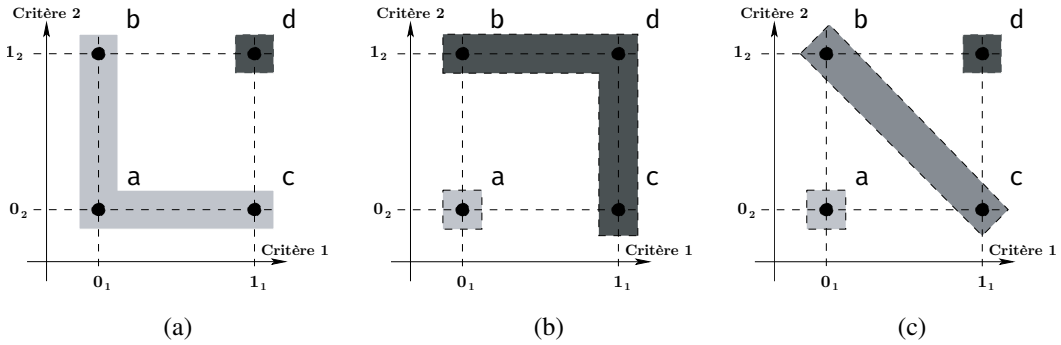


FIGURE 2.3 – Interaction entre deux critères ; d'après [74]

2.3.4 Simplification du modèle

L'objectif de cette section est de simplifier le modèle (en dégradant le moins possible son pouvoir expressif) afin de ne pas avoir à exprimer les $2^p - 2$ paramètres nécessaires à la description de μ. Les notions de *transformation de Möbius* et de *capacités k-additives* sont introduites à cette fin.

Définition 2.9 - Transformation de Möbius [122]
La transformation de Möbius $m : \mathfrak{P}(P) \to \mathbb{R}$ de la capacité μ est définie par :

$$m(A) = \sum_{K \subseteq A} (-1)^{|A \setminus K|} \, \mu(K)$$

L'expression de l'indice d'interaction d'un ensemble de critères (et par conséquent celle de la valeur de Shapley) peut être grandement simplifiée à l'aide de la transformation de Möbius [69] :

$$I(A) = \sum_{B \subseteq P \setminus A} \frac{1}{|B| + 1} \times m(A \cup B) \tag{2.1}$$

Définition 2.10 - K-additivité [70]
Une capacité μ est dite k-additive si sa transformation de Möbius vérifie :

- $\forall A \in \mathfrak{P}(P),\ |A| > k \ \Rightarrow\ m(A) = 0$
- $\exists A \in \mathfrak{P}(P),\ |A| = k$ et $m(A) \neq 0$

Les capacités k-additives sont particulièrement intéressantes car elles peuvent être exprimées à l'aide de seulement $\sum_{i=1}^{k} \binom{p}{i}$ paramètres. Dans la littérature, les capacités 2-additives sont souvent considérées comme l'un des meilleurs compromis entre expressivité et complexité du modèle [73]. En effet, seuls $\frac{p\,(p+1)}{2}$ paramètres sont nécessaires à leur description et elles permettent de représenter des interactions entre paires de critères (les interactions entre des ensembles de critères plus grands étant de toute façon complexes à appréhender pour le décideur).

Par mesure de simplicité, les notations ϕ_i, m_i, m_{ij} et I_{ij} sont utilisées à la place de $\phi(i)$, $m(\{i\})$, $m(\{i,j\})$ et $I(\{i,j\})$ dans la suite du document. De plus, $\wedge$ et $\vee$ désignent respectivement les opérateurs min et max. Ainsi, pour une capacité 2-additive, l'équation 2.1 permet d'écrire :

$$I_{ij} = m_{ij} \tag{2.2}$$

$$\phi_i = m_i + \frac{1}{2}\sum_{i \neq j} I_{ij} \tag{2.3}$$

L'intégrale de Choquet d'une capacité 2-additive peut alors être exprimée à partir des valeurs de Shapley ϕ_i et des indices d'interactions I_{ij}.

Définition 2.11 - Intégrale de Choquet 2-additive [73]
Soit un vecteur $a \in \mathbb{R}^p$ et μ une fonction de capacité définie sur P. L'intégrale de Choquet 2-additive $C_\mu : \mathbb{R}_+^p \to \mathbb{R}_+$ est définie par :

$$C_\mu(a) = \sum_{I_{ij}>0} (a_i \wedge a_j) \times I_{ij} + \sum_{I_{ij}<0} (a_i \vee a_j) \times |I_{ij}| + \sum_{i \in P} a_i \times \Big[\, \phi_i - \frac{1}{2}\sum_{j \neq i} |I_{ij}| \,\Big]$$

Sous cette forme, l'interprétation de l'intégrale de Choquet 2-additive est facilitée. Si l'indice d'interaction entre critères I_{ij} est positif, les critères sont agrégés par l'opérateur min puisqu'ils sont complémentaires. En revanche, lorsque l'indice d'interaction I_{ij} est négatif, les critères sont agrégés par l'opérateur max puisqu'ils sont substituables l'un par rapport à l'autre. Le troisième terme de l'expression constitue la partie linéaire de l'intégrale de Choquet et n'est autre que la somme pondérée des valeurs de Shapley ϕ_i à laquelle la somme des interactions relatives au critère i a été retranchée.

Par réécriture de la définition 2.11, l'intégrale de Choquet peut s'exprimer uniquement en fonction de la capacité m. Cette forme bien que moins intuitive que la précédente évite le calcul de ϕ_i et facilite ainsi l'implémentation de C_μ.

$$C_\mu(a) = \sum_{m_{ij}>0} (a_i \wedge a_j) \times m_{ij} + \sum_{m_{ij}<0} (a_i \vee a_j) \times |m_{ij}| + \sum_{i \in P} a_i \times \Big[m_i - \sum_{\substack{j \neq i \\ m_{ij}<0}} |m_{ij}| \Big]$$

Pour conclure, il convient de préciser que certaines préférences ne peuvent être représentées avec le modèle proposé (voir [71] par exemple). Ces dernières peuvent néanmoins être traitées en utilisant une échelle de satisfaction ξ bipolaire (qui généralise la notion d'échelle unipolaire) [73]. Il faut toutefois employer des modèles qui requièrent plus d'informations préférentielles et qui sont plus complexes à réaliser.

2.3.5 Retour sur l'exemple illustratif

L'exemple de la comparaison d'étudiants présentés à la section 2.2.4 est reconsidéré. Les notes des étudiants et les fonctions d'utilités partielles du décideur demeurent inchangées mais la fonction d'agrégation utilisée est à présent une intégrale de Choquet 2-additive.

La valeur des critères mathématique (M), physique (P) et langue (L) obtenue par chaque étudiant est rappelée dans le tableau suivant.

	Mathématique (M)	Physique (P)	Langue (L)
Etudiant E_a	0.75	0.75	0.75
Etudiant E_b	0.875	0.875	0.5
Etudiant E_c	0.75	0.625	0.75

TABLEAU 2.4 – Exemple : Notes des étudiants

Les préférences du décideur sont modélisées par une capacité μ choisie telle que l'importance d'un critère k seul soit égale au poids w_k de ce dernier dans l'exemple initial (cf. section 2.4.2 pour plus de détails sur la construction d'une fonction de capacité).

$\mathfrak{P}(P)$	$\emptyset$	M	P	L	MP	ML	PL	MPL
μ	0	0.3	0.3	0.4	0.4	0.8	0.8	1
m	0	0.3	0.3	0.4	-0.2	0.1	0.1	0

TABLEAU 2.5 – Exemple : Fonction de capacité considérée

La transformation de Möbius de la capacité 2-additive μ (troisième ligne de la figure 2.5) est obtenue à partir de l'équation 2.1. Dans le cas 2-additif, l'indice d'interaction d'une paire de critères est égal à son coefficient de Möbius (cf. équation 2.2) ce qui permet de dire que les critères M et P sont substituables alors que les critères M et L (respectivement P et L) sont complémentaires. Cette fonction de capacité traduit la préférence du décideur pour les étudiants au profil équilibré : redondance entre les notes de sciences et complémentarité de la note de langue avec les notes de mathématiques et de physique.

A titre d'exemple, l'utilité de E_b (respectivement de E_c) est calculée avec la définition 2.6 et μ (respectivement la définition 2.11 et m) :

$$\begin{aligned}
C_\mu(y^b) &= \mu(\{\sigma(M), \sigma(P), \sigma(L)\}) \times [\, y^b_{\sigma(M)} - y^b_{\sigma(0)} \,] + \\
&\quad \mu(\{\sigma(P), \sigma(L)\}) \times [\, y^b_{\sigma(P)} - y^b_{\sigma(M)} \,] + \\
&\quad \mu(\{\sigma(L)\}) \times [\, y^b_{\sigma(L)} - y^b_{\sigma(P)} \,] \\
&= y^b_L \; + \mu(\{M, P\}) \times [\, y^b_P - y^b_L \,] \; + \mu(\{M\}) \times [\, y^b_M - y^b_P \,] \\
&= 1 \times 0.5 \; + \; 0.4 \times 0.375 \; + \; 0.3 \times 0 \\
&= 0.65
\end{aligned}$$

$$\begin{aligned}
C_\mu(y^c) &= (y^c_M \wedge y^c_L) \times m_{ML} \; + \; (y^c_P \wedge y^c_L) \times m_{PL} \; + \\
&\quad (y^c_M \vee y^c_P) \times |m_{MP}| \; + \; y^c_M \times [\, m_M - |m_{MP}| \,] \; + \\
&\quad y^c_P \times [\, m_P - |m_{MP}| \,] \; + \; y^c_L \times m_L \\
&= 0.75 \times 0.1 \; + \; 0.625 \times 0.1 \; + \; 0.75 \times 0.2 \; + \\
&\quad 0.75 \times 0.1 \; + \; 0.625 \times 0.1 \; + \; 0.75 \times 0.4 \\
&= 0.725
\end{aligned}$$

Ainsi, les résultats obtenus lorsque C_μ est utilisée sont les suivants :

	Etudiant E_a	Etudiant E_b	Etudiant E_c
Utilité	0.75	0.65	0.725

TABLEAU 2.6 – Exemple : Utilités calculées à l'aide d'une intégrale de Choquet

Au sens de ce modèle MAUT construit à partir de u_{MP}, u_L et C_μ, les préférences du directeur au regard de ses étudiants sont : $E_a \succsim E_c \succsim E_b$. Il convient de noter que ce modèle représente mieux les préférences du directeur que celui basé sur la somme pondérée puisqu'ici l'étudiant E_c (profil équilibré) est préféré à l'étudiant E_b (performance en sciences uniquement). En outre, puisque la capacité a été choisie de sorte que $\mu(k) = w_k$, il apparait clairement que ce gain en pouvoir expressif provient de l'information supplémentaire que l'intégrale de Choquet peut modéliser à savoir les interactions entre critères.

Domaine de la gestion de crise

Les critères MAUT *peuvent être considérés comme des points de vue selon lesquels une solution est analysée. Il est possible d'agréger ces derniers les uns aux autres afin de déterminer le score associé aux solutions du problème considéré. Leur grande expressivité est particulièrement intéressante pour le domaine de la gestion de crise puisqu'ils peuvent être utilisés pour modéliser des préférences relativement variées. A titre d'exemple, le choix de la ministre de la santé française d'acquérir en masse et de façon préventive des vaccins contre la grippe H1N1 avait été fortement critiqué en 2009 en raison de son fort coût économique. En revanche, analysé selon un critère visant à maximiser les mesures de précaution, cette décision peut apparaître comme satisfaisante.*

Les préférences MAUT *sont par nature relativement différentes des préférences* PDDL3 *utilisées en planification. Il reste donc à déterminer si ces deux formalismes peuvent cohabiter l'un avec l'autre. La réponse à cette interrogation est affirmative comme expliqué dans la section 3.1 qui décrit une solution pour encoder des préférences* PDDL3 *à l'aide de critères* MAUT.

2.4 Réalisation d'un modèle MAUT couplé à une intégrale de Choquet

Cette section s'intéresse à la résolution du problème de l'*élicitation des préférences* qui consiste à construire un modèle des préférences du décideur. L'approche considérée ici est une extension de la méthode MACBETH [10] dans le cas de l'utilisation d'une intégrale de Choquet [93]. Pour modéliser les préférences du décideur, il est nécessaire de définir une fonction d'utilité u_k pour chaque critère k ainsi que l'intégrale de Choquet C_μ utilisée comme fonction d'agrégation ψ.

2.4.1 Construction des fonctions d'utilité partielles

Afin de construire les fonctions d'utilité $u_k : \Omega_k \to \xi$, il convient de préciser les deux *points de saturation* $\mathbf{1}_k$ et $\mathbf{0}_k$ de k sur l'échelle unipolaire ξ (cf. section 2.2.2). Ces deux éléments de Ω_k sont facilement identifiables puisqu'ils correspondent respectivement à une satisfaction complète et un insatisfaction totale du décideur. Pour le critère « note de mathématique » présenté dans la section 2.2.4, $\mathbf{1}_M = 20$ et $\mathbf{0}_M = 8$. Ceci signifie que du point de vue du décideur, une note de 6 en mathématiques est tout aussi insatisfaisante qu'une note de 8. Par convention, l'utilité des niveaux de saturation est fixée telle que $u_k(\mathbf{1}_k) = 1$ et $u_k(\mathbf{0}_k) = 0$.

Une fois que les niveaux de saturation ont été renseignés, il est également possible de considérer des *points de références* $\omega \in [\mathbf{0}_k\ ;\ \mathbf{1}_k]$. Par exemple, les points $\omega_M^1 = 12$ et $\omega_M^2 = 16$ sont les deux points de référence de la fonction d'utilité u_M de la section 2.2.4. Il est très difficile pour les décideurs de donner des informations préférentielles absolues (ou cardinales). En conséquence, il n'est pas raisonnable de leur demander les valeurs de $u_k(\omega)$ directement. En revanche, les décideurs peuvent exprimer plus facilement des informations préférentielles ordinales telles que des différences d'utilité entre deux points de référence par exemple. La méthode MACBETH [10] décrit une procédure de questionnement qui permet de déterminer l'utilité associée à chaque point de référence sur la base d'informations préférentielles ordinales. Cette dernière est étendue dans [93] afin de prendre en considération d'éventuelles interactions entre critères. Afin de décrire cette approche, l'ensemble $X_k = \{\, (\omega_k, \mathbf{0}_{-k}) \mid \omega_k \in \Omega_k \,\}$ est introduit. Les éléments de X_k sont des alterna-

tives hypothétiques qui seraient totalement insatisfaisantes sur tous les critères à l'exception du critère k dont la valeur serait ω_k. Au cours de la procédure de questionnement, les décideurs comparent des paires d'alternatives $x_k^a, x_k^b \in X_k$. Ils doivent préciser si x_k^a et x_k^b sont jugées indifférentes ($x_k^a \sim x_k^b$) ou si x_k^a est préférée à x_k^b ($x_k^a \succ x_k^b$) et définir le cas échéant la différence de satisfaction entre x_k^a et x_k^b comme étant *très faible* (C_1), *faible* (C_2), *modérée* (C_3), *forte* (C_4), *très forte* (C_5) ou *extrême* (C_6). La figure 2.7 illustre les réponses du décideur à la procédure de questionnement utilisée pour construire la fonction d'utilité u_M.

X_M	$(8_M, \mathbf{0}_{-M})$	$(12_M, \mathbf{0}_{-M})$	$(16_M, \mathbf{0}_{-M})$	$(20_M, \mathbf{0}_{-M})$
$(8_M, \mathbf{0}_{-M})$	$\sim$	C_1	C_5	C_6
$(12_M, \mathbf{0}_{-M})$		$\sim$	C_4	C_5
$(16_M, \mathbf{0}_{-M})$			$\sim$	C_1
$(20_M, \mathbf{0}_{-M})$				$\sim$

TABLEAU 2.7 – Illustration d'une procédure de questionnement sur X_M

Les informations extraites de la procédure de questionnement sont en partie redondantes et des inconsistances peuvent être introduites par les choix du décideur. Par exemple, pour trois alternatives $x_k^a, x_k^b, x_k^c \in X_k$, si le décideur déclare préférer x_k^a à x_k^b, x_k^b à x_k^c et x_k^c à x_k^a, alors son choix est inconsistant puisqu'il introduit un cycle dans la structure de la relation de préférence $x_k^a \succ x_k^b \succ x_k^c \succ x_k^a$. De même, si $x_k^a \succ x_k^b \succ x_k^c$ avec la différence de satisfaction entre x_k^a et x_k^b considérée comme *très faible* et la différence de satisfaction entre x_k^b et x_k^c également jugée *très faible*, alors considérer la différence de satisfaction entre x_k^a et x_k^c comme *extrême* constituerait un choix inconsistant de la part du décideur. Plus généralement, la notion de cyclone [9] permet de caractériser les inconsistances dans la structure de la relation de préférence induite par les choix des décideurs. Ainsi, ces dernières peuvent être détectées automatiquement et il est possible d'expliquer au décideur pourquoi certains de ses choix sont en conflit entre eux. Dès lors que les informations préférentielles du décideur sont consistantes, elles sont utilisées pour définir un problème de programmation linéaire (voir [8] pour son expression exacte). La résolution de ce problème détermine l'utilité de chaque point de référence ω.

A ce stade, l'utilité des points de saturation et des points de référence est connue. En utilisant ces points comme les bornes z_k^i de la définition 2.3, il est possible de déterminer l'utilité de n'importe quel point de Ω_k par interpolation linéaire. Ainsi, la fonction d'utilité partielle u_k est entièrement définie. Le logiciel MYRIAD [95] implémente l'approche décrite (définition des points de saturation, procédure de questionnement, calcul de l'utilité des points de référence et interpolation linéaire) et peut être utilisé pour construire rapidement les fonctions d'utilité u_k.

2.4.2 Construction de la fonction d'agrégation

Une fois que toutes les fonctions d'utilité u_k ont été précisées, il faut construire une capacité 2-additive μ. Cette dernière définit l'intégrale de Choquet utilisée comme fonction d'agrégation ψ. Comme dans le cas des fonctions d'utilité u_k, les valeurs de la capacité μ sont déterminées à l'aide d'une procédure de questionnement portant sur un sous-ensemble des alternatives X.

Pour rappel, $X_A = \{ (\mathbf{1}_A, \mathbf{0}_{-A}) \mid A \subseteq P \}$ est l'ensemble des *alternatives binaires*. Les alternatives binaires sont des alternatives hypothétiques qui seraient complètement satisfaisantes sur les critères $k \in A$ et totalement insatisfaisantes sur les autres critères. Etant donné que $u_k(\mathbf{1}_k) = 1$ et $u_k(\mathbf{0}_k) = 0$, l'utilité d'une alternative binaire se réduit à $C_\mu(\mathbf{1}_A, \mathbf{0}_{-A})$ qui est égal à $\mu(A)$ par définition. En conséquence, les valeurs de la capacité 2-additive μ peuvent être obtenues à partir des informations préférentielles du décideur sur l'ensemble des alternatives binaires X_A. Pour cela, une procédure de questionnement analogue à celle présentée dans la section 2.4.1 est mise en œuvre. Il est demandé aux décideurs de comparer des paires d'alternatives binaires $x_A^a, x_A^b \in X_A$. Pour chaque paire d'alternatives, ils doivent préciser si x_A^a et x_A^b sont jugées indifférentes ($x_A^a \sim x_A^b$) ou si x_A^a est préférée à x_A^b ($x_A^a \succsim x_A^b$) auquel cas ils doivent caractériser la différence de satisfaction entre x_A^a et x_A^b à l'aide des six catégories suivantes : *très faible* (C_1), *faible* (C_2), *modérée* (C_3), *forte* (C_4), *très forte* (C_5) ou *extrême* (C_6). La figure 2.8 illustre les réponses du décideur à la procédure de questionnement utilisée pour construire la capacité μ présentée dans l'exemple de la section 2.3.5.

X_A	$(\mathbf{1}_M, \mathbf{0}_{-M})$	$(\mathbf{1}_P, \mathbf{0}_{-P})$	$(\mathbf{1}_L, \mathbf{0}_{-L})$	$(\mathbf{1}_{MP}, \mathbf{0}_{-MP})$	$(\mathbf{1}_{ML}, \mathbf{0}_{-ML})$	$(\mathbf{1}_{PL}, \mathbf{0}_{-PL})$
$(\mathbf{1}_M, \mathbf{0}_{-M})$	$\sim$	$\sim$	C_1	C_1	C_5	C_5
$(\mathbf{1}_P, \mathbf{0}_{-P})$		$\sim$	C_1	C_1	C_5	C_5
$(\mathbf{1}_L, \mathbf{0}_{-L})$			$\sim$	$\sim$	C_4	C_4
$(\mathbf{1}_{MP}, \mathbf{0}_{-MP})$				$\sim$	C_4	C_4
$(\mathbf{1}_{ML}, \mathbf{0}_{-ML})$					$\sim$	$\sim$
$(\mathbf{1}_{PL}, \mathbf{0}_{-PL})$						$\sim$

TABLEAU 2.8 – Illustration d'une procédure de questionnement sur X_A

Comme dans le cas des fonctions d'utilité partielles u_k, les choix du décideur peuvent introduire d'éventuelles inconsistances. Ces dernières peuvent être illustrées (exemple emprunté à [104]) à l'aide d'un ensemble de trois critères $P = \{1, 2, 3\}$ et de six alternatives binaires $x_\emptyset, x_1, x_2, x_3, x_{12}, x_{13} \in X_A$ telles que $x_{12} \sim x_3$, $x_{13} \sim x_2$ et $x_1 \succ x_\emptyset$. Considérées conjointement avec la propriété de monotonicité de la capacité qui impose $x_{12} \succsim x_2$ et $x_{13} \succsim x_3$, ces préférences impliquent nécessairement $\mu_{12} = \mu_{13} = \mu_2 = \mu_3$. En outre, $x_1 \succ x_\emptyset$ impose $\mu_1 > 0$ ce qui entre en contradiction avec la contrainte de monotonicité $\mu_{12} + \mu_{13} \geq \mu_1 + \mu_2 + \mu_3$ (voir [70] pour l'expression formelle des contraintes de monotonicité qu'une capacité doit respecter). La condition MOPI [105] permet de caractériser et donc de détecter ce type d'inconsistances. Des conseils peuvent alors être prodigués au décideur afin que ce dernier modifie ses choix en conséquence. Une fois que les informations préférentielles sont consistantes, un algorithme dérivé des travaux présentés dans [104] peut être mis en œuvre afin de construire la fonction de capacité μ. C'est ce que propose le logiciel MYRIAD [95] qui peut définir rapidement une intégrale de Choquet 2-additive à partir des informations préférentielles d'un décideur sur l'ensemble des alternatives binaires.

En utilisant les méthodes exposées dans les sections 2.4.1 et 2.4.2, les fonctions d'utilités partielles u_k et la fonction d'agrégation ψ d'un modèle MAUT peuvent être construites. La fonction d'utilité U du modèle est alors entièrement caractérisée ce qui permet de préciser la relation d'ordre des décideurs (cf. définition 2.2).

Domaine de la gestion de crise
Les décideurs de la gestion de crise sont des experts opérationnels. La démarche de construction présentée dans cette section est donc particulièrement importante dans le domaine de la gestion de crise puisqu'elle permet aux décideurs de modéliser leurs préférences uniquement à partir d'informations préférentielles opérationnelles. Ainsi, il ne leur est jamais demandé de préciser les paramètres mathématiques du modèle de préférences mais uniquement d'effectuer des comparaisons opérationnelles entre plusieurs plans hypothétiques.

Conclusion

Après une brève présentation des problèmes traités en aide à la décision multicritère, ce chapitre s'est focalisé sur la question de la modélisation des préférences du décideur. Cette étape fondamentale pour la résolution des problèmes d'aide à la décision multicritère a été adressée dans le cadre de la Théorie de l'Utilité Multi-Attribut (MAUT). Ce formalisme propose de modéliser chacune des préférences du décideur à l'aide d'un attribut associé à une fonction d'utilité partielle. Par la suite, une fonction d'agrégation est utilisée pour déterminer l'utilité d'une alternative sur la base des utilités partielles associées aux différentes préférences du décideur. L'opérateur d'agrégation considéré dans ce chapitre est l'intégrale de Choquet. Cette dernière possède un grand pouvoir expressif puisqu'elle permet notamment de représenter des interactions entre les préférences du décideur. Cette richesse s'accompagne malheureusement d'une grande complexité ce qui a motivé l'étude du cas particulier des intégrales de Choquet 2-additives qui sont souvent considérées comme le meilleur compromis entre expressivité et complexité. En effet, leur caractérisation ne requiert qu'un faible nombre de paramètres mais elles peuvent néanmoins modéliser des interactions entre paires de critères. Après avoir présenté en détails le formalisme MAUT et l'intégrale de Choquet, la question de la construction d'un modèle de préférences basé sur ces éléments a été traitée. Les éléments présentés ici sont exploités dans le chapitre 3 qui propose de résoudre le problème de la planification avec préférences en représentant les préférences à l'aide du formalisme MAUT.

Chapitre 3

Planification avec préférences basée sur la théorie MAUT couplée à une intégrale de Choquet

Ce chapitre s'intéresse à la planification avec préférences dans le cas où les préférences sont formalisées à l'aide d'un modèle MAUT et d'une intégrale de Choquet. Par ailleurs, il introduit également le planificateur ChoPlan.

La section 3.1 présente le formalisme PDDL3/MAUT qui permet de représenter les préférences PDDL3 à l'aide de critères MAUT et d'utiliser une intégrale de Choquet comme fonction objectif. Lorsque ce formalisme est utilisé, le décideur dispose d'un plus grand pouvoir expressif quant à l'expression de ses préférences dans les problèmes de planification. La section 3.2 s'intéresse quant à elle à la résolution des problèmes de planification avec préférences représentés selon le formalisme PDDL3/MAUT. Un algorithme générique de recherche guidée par une heuristique est proposé. Celui-ci doit être instancié à l'aide d'une règle de sélection et d'un ensemble (éventuellement vide) de règles de coupe. Six règles de sélection candidates et une règle de coupe sont suggérées. Ces dernières sont utilisées par le planificateur ChoPlan dont l'implémentation est décrite dans la section 3.3. Pour finir, les performances de ChoPlan sont évaluées par rapport aux planificateurs de l'art en utilisant un ensemble de problèmes issus des compétitions internationales de planification.

3.1 Extension du pouvoir expressif du langage PDDL3

Cette section présente un formalisme original pour la planification avec préférences. Déjà mentionné dans la section 1.2.3 sous l'appellation PDDL3/MAUT, celui-ci généralise la notion de préférence utilisée en PDDL3 par l'introduction de préférences floues. L'extension proposée élargit le périmètre des problèmes de planification avec préférences qu'il est possible de modéliser et de résoudre. Pour réaliser cette généralisation, les préférences PDDL sont encodées en tant que critères MAUT (cf. section 3.1.1) et la fonction objectif PDDL est définie à l'aide d'une intégrale de Choquet (cf. section 3.1.2). Les modifications à apporter au langage PDDL pour prendre en compte ces changements sont présentés dans la section 3.1.3.

3.1.1 Préférences PDDL3 et critères MAUT

Trois types de préférences ont été introduites dans la section 1.2.1 à savoir les préférences numériques, finales et de trajectoire. De plus, il a été expliqué dans la section 2.2 qu'un critère MAUT est modélisé par un attribut auquel est associée une fonction d'utilité partielle. Cette dernière représente les préférences du décideur par rapport aux valeurs de l'attribut. Cette section explique comment des critères MAUT peuvent être utilisés pour représenter les préférences d'un problème de planification.

Le cas des préférences numériques est le plus simple puisque ces dernières définissent intrinsèquement l'attribut à considérer. Il ne reste donc qu'à définir la fonction d'utilité partielle de l'attribut. Celle-ci peut être construite manuellement ou à l'aide de la méthode présentée dans la section 2.4.2. Par exemple, dans le problème *Rovers* [42], le décideur peut avoir envie de raisonner sur la quantité d'énergie consommée par le robot $N1$ comme mentionné dans la section 1.1.2. La variable numérique e_1 associée à la consommation d'énergie du robot $N1$ constitue naturellement l'attribut de cette préférence. Le décideur peut être complètement satisfait (respectivement totalement insatisfait) si 40 unités d'énergie (respectivement 100 unités) sont consommées lors de la mission. Par ailleurs, sa satisfaction n'évolue pas nécessairement linéairement et il peut fortement préférer une consommation de 80 unités par rapport à une de 100 unités mais ne préférer que modérément une consommation de 40 unités par rapport à une de 60 unités. Ces informations

préférentielles peuvent être utilisées pour construire une fonction d'utilité partielle $u_{EC} : [0, 120] \rightarrow [0, 1]$ représentant les choix du décideur comme illustré sur la figure 3.1. En conséquence, la préférence numérique du décideur quant à la consommation d'énergie du robot $N1$ peut être définie par l'attribut e_1 et la fonction d'utilité partielle u_{EC}. Ainsi, si dans un plan x, le robot $N1$ consomme 60 unités d'énergie au cours de sa mission alors l'utilité de la préférence dans ce plan est $u_{EC}(60) = 0.8$. Ceci s'interprète comme une grande satisfaction du décideur quant à la consommation d'énergie du robot $N1$ dans x.

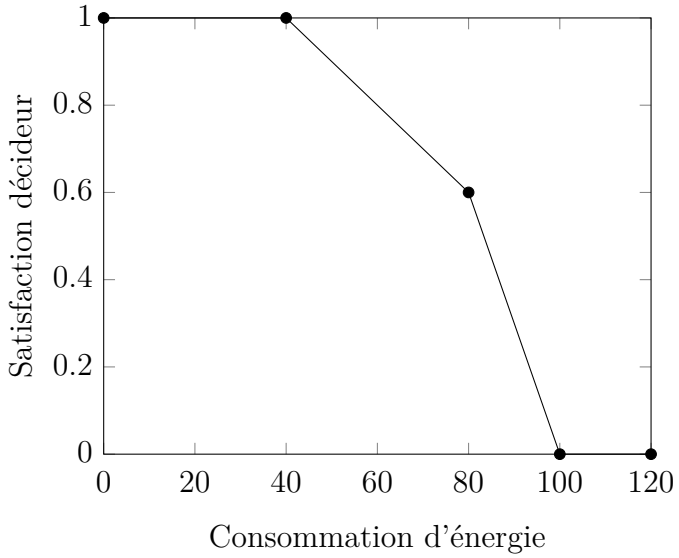


FIGURE 3.1 – Fonction d'utilité partielle u_{EC}

Le cas des préférences finales et des préférences de trajectoire est moins intuitif mais reste néanmoins simple à traiter. En effet, il suffit de considérer un attribut dont la valeur est définie par la sémantique de la préférence considérée (cf. définition 1.24). Ainsi, l'attribut vaut 1 si la préférence est satisfaite par rapport à la trajectoire considérée et 0 dans le cas contraire. Afin de respecter la nature binaire des préférences PDDL3, la fonction d'utilité partielle $u : \{0, 1\} \rightarrow [0, 1]$ associée à l'attribut est définie telle que $u(0) = 0$ et $u(1) = 1$. A titre d'illustration, dans le problème *Rovers*, la préférence `preference (always (at N1 L1))` signifie qu'il est préférable que le robot $N1$ soit toujours dans le lieu $L1$. Si dans un plan donné, le robot $N1$ se trouve dans une autre zone, alors la préférence est violée et son attribut prend pour valeur 0. En conséquence, la valeur d'utilité partielle associée à cette préférence dans le plan considéré est également nulle.

Le mécanisme considéré pour encapsuler les préférences finales et les préférences de trajectoire du PDDL3 permet de représenter ces dernières par un critère MAUT tout en conservant leur sémantique initiale. Cette méthode peut sembler relativement pauvre de prime abord puisqu'en respectant la sémantique du PDDL3, elle ne profite pas pleinement des avantages liés à la nature floue des critères MAUT ; néanmoins il n'en est rien. Tout d'abord, ce choix permet au formalisme PDDL3/MAUT d'être pleinement compatible avec la sémantique du PDDL3. Ce dernier s'inscrit donc dans la continuité des travaux réalisés autour du langage PDDL et ne constitue pas une alternative à la représentation de référence qu'est le PDDL3. Ainsi, tout planificateur capable d'utiliser le langage PDDL3 peut être étendu pour supporter l'extension PDDL3/MAUT. De plus, ce mécanisme d'encapsulation rend les préférences finales et les préférences de trajectoires commensurables avec les préférences numériques ce qui est particulièrement intéressant lors de la création de la fonction objectif comme expliqué dans la section 3.1.2. Pour finir, il convient de préciser que la nature floue de ces préférences peut néanmoins être exploitée pour élaborer des heuristiques comme présenté dans la section 3.2.

Les trois types de préférences considérées jusque là peuvent être vues comme des *préférences élémentaires* dans la mesure où elles ne s'intéressent qu'à un seul attribut des plans. Le langage PDDL3/MAUT permet également de représenter des *préférences non élémentaires* formées par agrégation d'un ensemble de préférences élémentaires. L'utilité d'une telle préférence est définie à l'aide d'une fonction d'agrégation évaluée avec les valeurs d'utilités des préférences élémentaires considérées. A titre d'illustration, dans un problème *Rovers* avec trois robots $N1$, $N2$ et $N3$, le décideur pourrait être intéressé par la création d'une préférence non élémentaire qui caractérise la consommation d'énergie engendrée par la mission sur la base des consommation d'énergie de chacun des robots. Soit x un plan ; p la préférence non élémentaire représentant la consommation d'énergie de la mission ; p_1, p_2, p_3 les trois préférences élémentaires agrégées par p qui représentent les consommations d'énergie individuelles des robots $N1$, $N2$ et $N3$; $y = z(x)$ le vecteur représentant le plan x dans l'espace des attributs $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3$ c'est à dire le vecteur des consommations des robots ; ν une capacité définie sur $\mathfrak{P}(P)$ où P est l'ensemble des attributs relatifs à p_1, p_2 et p_3 ; et C_ν une intégrale de Choquet qui agrège les consommations individuelles des robots pour obtenir la consommation d'énergie de

la mission. L'utilité associée à la préférence non élémentaire p dans le plan x est définie par $C_\nu(u_1(y_1), u_2(y_2), u_3(y_3))$.

Pour finir, il convient de préciser qu'il est possible d'ajouter d'autres types de préférences élémentaires dans le formalisme PDDL3/MAUT. En effet, ce dernier possède par nature un grand pouvoir expressif puisque les critères MAUT peuvent représenter n'importe quelle variable. Par exemple, un critère pourrait dénombrer les états d'un plan dans lesquels une formule ϕ n'est pas vérifiée ; ce dernier constituerait alors une interprétation floue des préférences PDDL3 de type `always`. L'ajout de nouveaux types de préférences élémentaires permet également de créer des mécanismes de préférences spécifiques à une classe de problème donnée. La description BNF du formalisme PDDL3/MAUT présentée dans la section 3.1.3 se limite aux préférences élémentaires de type finale, de trajectoire ou numérique.

3.1.2 Fonction objectif PDDL et intégrale de Choquet

Une fois que des préférences ont été spécifiées, il faut se munir d'un mécanisme permettant de calculer la qualité (appelée utilité dans la terminologie MAUT) des plans solutions. En PDDL3, un coût de violation est associé à chaque préférence puis une fonction objectif est définie à partir de ces derniers. Dans le cas de l'extension PDDL3/MAUT, la fonction objectif est construite à l'aide d'une fonction d'agrégation comme le suggère la définition 2.2.

L'agrégation des préférences est plus simple et plus intuitive dans le formalisme PDDL3/MAUT puisque le modèle MAUT impose la *commensurabilité* des différents critères considérés (c.-à-d. la possibilité de les comparer par le biais d'une unité adéquate). En effet, il est relativement facile de comparer plusieurs préférences entre elles puisque ces dernières sont toutes définies en utilisant une fonction d'utilité partielle à valeur sur l'échelle de satisfaction commune ξ. Les problèmes qui mettent en oeuvre beaucoup de préférences numériques peuvent donc être encodés sans difficulté avec le formalisme MAUT. Ce propos est illustré une fois de plus à l'aide du problème *Rovers*. Si les robots peuvent être endommagés lorsqu'ils prélèvent un échantillon, le décideur peut introduire un critère qui modélise le risque d'endommagement des équipements des robots. L'espace de définition associé à ce risque est l'ensemble discret $\Omega_R = \{$`Très faible`, `Faible`, `Modéré`, `Elevé`, `Très élevé`$\}$.

Sans commensurabilité, il est impossible de déterminer si une amélioration sur le critère de consommation d'énergie (par exemple consommer 50 unités plutôt que 70) est préférée à une amélioration sur le critère de risque (par exemple passer d'un risque `Elevé` à un risque `Modéré`). En conséquence, construire une fonction objectif qui agrège les préférences du décideur relatives à la consommation d'énergie du robot $N1$, le risque de casse de ses équipements et la préférence de trajectoire `preference (always (N1 L1))` est difficile en PDDL3. Ceci explique notamment pourquoi tous les problèmes de planification avec préférences proposés lors des compétitions IPC n'utilisent au plus qu'une seule préférence numérique. En revanche, lorsque les préférences sont représentées par des critères MAUT, il est plus facile de les agréger puisqu'elles sont toutes commensurables entre elles. En effet, la notion d'utilité modélise la même quantité quelque soit la préférence considérée à savoir la satisfaction du décideur. Ainsi, le formalisme PDDL3/MAUT permet de représenter des problèmes avec un grand nombre de préférences numériques.

Bien que la fonction objectif PDDL puisse être quelconque, les problèmes de référence utilisent généralement une somme pondérée. L'opérateur d'agrégation retenu dans le cadre de cette étude est l'intégrale de Choquet 2-additive. Celle-ci généralise la somme pondérée en permettant au décideur de représenter des interactions entre paires de critères. A titre d'illustration, un décideur peut être prêt à accepter une grande consommation d'énergie pour un robot si le risque d'endommagement de ses équipements est faible. Dans ce cas, les deux critères sont substituables et leur indice d'interaction est négatif (cf. section 2.3.3). Le langage PDDL3/MAUT permet donc de représenter plus finement la complexité intrinsèque des préférences du décideur.

L'extension PDDL3/MAUT améliore le pouvoir expressif du modèle de préférences considéré mais complexifie également la réalisation de ce dernier. Ceci est notamment dû au fait que les décideurs doivent définir une fonction de capacité. Toutefois, la construction de la capacité peut être automatisée et outillée comme expliqué dans la section 2.4.2. Ainsi, la réalisation d'un modèle de préférences PDDL3/MCDA peut être considérée plus simple que celle d'un modèle de préférences PDDL3. En effet, il est plus intuitif pour les décideurs de comparer des alternatives de solutions entre elles (PDDL3/MAUT) que de définir les paramètres mathématiques d'une somme pondérée (PDDL3).

3.1.3 Langage formel pour l'extension PDDL3/MAUT

Cette section présente les éléments syntaxiques et sémantiques à considérer pour étendre le langage PDDL3 à l'aide du formalisme MAUT.

Syntaxe de l'extension PDDL3/MAUT

Le pouvoir expressif du PDDL étant très vaste, peu de planificateurs implémentent l'ensemble des fonctionnalités supportées par ce dernier. Les *exigences* (nommées `requirement`) sont des sous-ensembles cohérents du langage PDDL qui ont été introduits afin de préciser les fonctionnalités qu'un planificateur peut mettre en œuvre. Par exemple, pour raisonner sur les expressions numériques et les préférences présentées respectivement dans les sections 1.1.3 et 1.2.3, il faut être capable de prendre en charge les éléments des exigences `numeric-fluents` et `preferences`.

Les modifications liées à l'extension PDDL3/MAUT sont accessibles par l'intermédiaire de l'exigence `maut-preferences` qui elle-même s'appuie sur les exigences `numeric-fluents` et `preferences`. La description BNF [4] qui spécifie l'exigence `maut-preferences` est présentée sur la figure 3.2. Cette dernière s'appuie sur la description BNF du PDDL qui est proposée dans [63]. Sans surprise, les deux éléments principaux de l'exigence `maut-preferences` sont les concepts de critères MAUT (`<maut-criterion>`) et d'intégrale de Choquet (`<choquet-integral>`). Les critères MAUT peuvent décrire des préférences finales ou des préférences de trajectoires (`<trajectory-criterion>`), des préférences numériques (`<numeric-criterion>`) ou encore des préférences non élémentaires (`<aggregation-criterion>`). Il convient de remarquer que d'un point de vue sémantique, les préférences finales s'interprètent sur une trajectoire (cf. définitions 1.23 et 1.24) ce qui justifie qu'elles soient représentées par un critère de type `<trajectory-criterion>`. L'intégrale de Choquet est quant à elle spécifiée à l'aide d'une capacité représentée par sa transformation de Möbius (`<mobius-capacity-list>`). La description proposée permet d'utiliser une intégrale de Choquet quelconque mais cette étude se limite au cas des intégrales de Choquet 2-additives.

```
<problem>                   ::= (...
                                  [<constraints>]:constraints
                                  [<maut-preferences>]:maut-preferences
                                  [<metric-spec>]:numeric-fluents
                                ...)


<maut-preferences>          ::=:maut-preferences (:maut-preferences <maut-spec>*)
<maut-spec>                 ::=:maut-preferences (:maut-criterion <maut-criterion>)
<maut-spec>                 ::=:maut-preferences (:choquet-integral <choquet-integral>)


<maut-criterion>            ::=:maut-preferences <numeric-criterion>
<maut-criterion>            ::=:maut-preferences <trajectory-criterion>
<maut-criterion>            ::=:maut-preferences <aggregation-criterion>

<numeric-criterion>         ::=:maut-preferences (:numeric-criterion <maut-criterion-symbol>
                                :attribute <basic-function-term>
                                :utility-function (<function-value>*))

<trajectory-criterion>      ::=:maut-preferences (:trajectory-criterion <maut-criterion-symbol>
                                :preference (<pref-name>))

<aggregation-criterion>     ::=:maut-preferences (:aggregation-criterion <maut-criterion-symbol>
                                :criteria (<maut-criterion-list>*)
                                :choquet-integral (<choquet-integral-symbol>))

<function-value>            ::=:maut-preferences (<number>, <number>)
<maut-criterion-list>       ::=:maut-preferences (<maut-criterion-symbol>)
<maut-criterion-symbol>     ::=:maut-preferences <name>


<choquet-integral>          ::=::maut-preferences (:choquet-integral <choquet-integral-symbol>
                                  :mobius (<mobius-capacity-list>*))

<mobius-capacity-list>      ::=:maut-preferences (<maut-criterion-symbol>
                                  [maut-criterion-symbol]* <number>)

<choquet-integral-symbol>   ::=:maut-preferences <name>


<metric-f-exp>              ::=:maut-preferences <choquet-integral-symbol>
```

FIGURE 3.2 – Description BNF de l'exigence PDDL `maut-preferences`

Les figures 3.3, 3.4 et 3.5 illustrent la syntaxe de l'extension PDDL3/MAUT à l'aide d'un problème *Rovers* construit à partir des exemple présentés dans les sections 1.1.2 et 1.2.2 (cf. annexe B.3 pour visualiser la totalité de l'exemple). Ce problème met en œuvre une préférence numérique (`c-e1`) relative à la consommation d'énergie du robot $N1$, une préférence finale (`c-f1`) relative à l'acquisition d'un échantillon de sol du lieu $L6$ et deux préférences de trajectoires (`c-s1` et `c-a1`) relatives à la position du robot $N1$ au cours de la mission.

```
(:functions
  (energy ?x - robot) - number
)
```

FIGURE 3.3 – Syntaxe d'expression numérique primitive en PDDL

```
(:objects
  N1 - robot   ...
)

(:goal (and ...
  (preference f1 (possede_echantillon_sol N1 L6))
  ))

(:constraints (and ...
  (preference s1 (sometime (position N1 L1)))
  (preference a1 (at-most-once (position N1 L6)))
))
```

FIGURE 3.4 – Syntaxe de préférences finales et de trajectoires en PDDL

```
(:maut-preferences
  (:numeric-criterion c-e1
    :attribute (energy N1)
    :utility-function (
      (40, 1)
      (80, 0.6)
      (100, 0)
    ))

  (:trajectory-criterion c-f1
    :preference (f1))

  (:trajectory-criterion c-s1
    :preference (s1))

  (:trajectory-criterion c-a1
    :preference (a1))

  (:choquet-integral choqInt
    :mobius (
      (c-e1 0.35)
      (c-f1 0.25)
      (c-s1 0.2)
      (c-a1 0.2)
      (c-e1 c-f1 -0.1)
      (c-s1 c-a1 0.1)
    )
  )
)

(:metric maximize choqInt)
```

FIGURE 3.5 – Syntaxe de l'extension PDDL3/MAUT

Sémantique de l'extension PDDL3/MAUT

Cette section présente la sémantique associée aux préférences lorsque ces dernières sont encodées dans des critères MAUT. Les définitions proposées s'appuient sur les éléments introduits dans les sections 1.1.3 et 1.2.3.

Définition 3.1 - Instance d'un problème de planification
Une instance d'un problème de planification avec préférences I_M est redéfinie par extension de la définition 1.14 en ajoutant la formule G', l'ensemble MC, la fonction objectif C_μ ainsi que la relation de préférences $\succsim$ à l'objet problème $Prob = (O, s_0, G', MC, C_\mu, \succsim)$ avec :

- G' une formule qui représente un ensemble d'objectifs étendus
- MC un ensemble de critères MAUT qui représentent des préférences
- C_μ une intégrale de Choquet 2-additive
- $\succsim$ une relation d'ordre construite à partir de MC et C_μ

Définition 3.2 - Utilité d'une préférence numérique
Soit t la trajectoire $\langle s_0, \cdots, s_n \rangle$ dont l'état final s_n est décrit par la paire (Atm, v), f une expression numérique primitive et $k \in MC$ le critère MAUT représentant la préférence numérique ayant pour attribut f et pour fonction d'utilité partielle $u_k : \Omega_f \to [0, 1]$. L'utilité associée à k sur la trajectoire t s'interprète par rapport à s_n et vaut $u_k^t = u_k(v_{Ind(f)})$.

Le problème *Rovers* présenté dans la section 1.2.2 (et dont la description exacte est donnée dans l'annexe B.4) est utilisé pour illustrer le calcul des utilités partielles associées à des préférences MAUT. L'une des solutions de ce problème est le plan 2 ; lequel est caractérisé par la trajectoire $t_2 = \langle s_0, \cdots, s_{10} \rangle$. L'exécution de ce plan conduit à un état final dans lequel la quantité d'énergie consommé par le robot $N1$ est de 50 unités (5 déplacements consommant chacun 10 unités d'énergie). Pour rappel, la fonction Ind permet d'associer à une expression numérique primitive f sa valeur dans les différents états du problème (cf. définition 1.16). Ainsi, la valeur associée au critère k relatif à la consommation d'énergie de $N1$ (nommé `c-e1` sur la figure 3.5) est celle de l'expression numérique primitive `(energy N1)` dans le vecteur v_{10} de s_{10}. L'utilité de ce critère est donc $u_{\texttt{c-e1}}^{t_2} = u_{EC}(50) = 0,9$ ce qui traduit une grande satisfaction du décideur quant à cette préférence.

Définition 3.3 - Utilité d'une préférence de trajectoire
Soit t la trajectoire $\langle s_0, \cdots, s_n \rangle$, Φ un objectif de trajectoire et $k \in MC$ le critère MAUT représentant la préférence de trajectoire associée à Φ ayant pour fonction d'utilité partielle $u_k : \{0,1\} \rightarrow [0,1]$. L'utilité associée à k sur la trajectoire t s'interprète par rapport à la valeur de vérité de Φ et vaut :

$$u_k^t = \begin{cases} 0 & \text{si} \quad \langle s_0, \cdots, s_n \rangle \models \neg\, \Phi \\ 1 & \text{si} \quad \langle s_0, \cdots, s_n \rangle \models \Phi \end{cases}$$

Les préférences de trajectoires `f1` et `a1` présentées sur la figure 3.4 sont vérifiées dans le plan 2. Les utilités des critères MAUT associés à ces préférences (nommés `c-f1` et `c-a1` sur la figure 3.5) valent donc donc $u^{t_2}_{\texttt{c-f1}} = u^{t_2}_{\texttt{c-a1}} = 1$. En revanche, l'utilité $u^{t_2}_{\texttt{c-s1}}$ du critère `c-s1` associé à la préférence `s1` est nulle puisque cette dernière est violée dans le plan 2 (le robot $N1$ ne se trouve jamais dans le lieu $L1$).

Définition 3.4 - Utilité d'une préférence non élémentaire
Soit t la trajectoire $\langle s_0, \cdots, s_n \rangle$; $k_1, \cdots, k_q$ des critères MAUT dont les valeurs d'utilité partielles sur t sont $u_1^t, \cdots, u_q^t$; ν une capacité ; et k un critère MAUT qui agrège les critères $k_1, \cdots, k_q$ à l'aide de l'intégrale de Choquet C_ν. L'utilité associée à k sur la trajectoire t vaut $u_k^t = C_\nu(u_1^t, \cdots, u_q^t)$.

Dans le cas de l'exemple de la section 1.2.2, le décideur pourrait souhaiter regrouper l'ensemble de ses préférences concernant la position du robot $N1$ (c.-à-d. `c-a1` et `c-s1`) en un unique critère `c-p`. Si la capacité ν est choisie de sorte à décrire une moyenne arithmétique, alors la valeur d'utilité du critère `c-p` dans le plan 2 est $u^{t_2}_{\texttt{c-p}} = C_\nu(u^{t_2}_{\texttt{c-a1}}\,,\, u^{t_2}_{\texttt{c-s1}}) = (u^{t_2}_{\texttt{c-a1}} + u^{t_2}_{\texttt{c-s1}})/2 = 0,5$ ce qui s'interprète comme une satisfaction moyenne du décideur.

Définition 3.5 - Utilité d'un plan
Soit I_M une instance d'un problème de planification, x un plan de trajectoire t et $u_1^t, \cdots, u_p^t$ les valeurs d'utilités partielles des critères MC de I_M sur t.

L'utilité U du plan x est définie telle que $U_x = C_\mu(u_1^t, \cdots, u_p^t)$. Le plan x_1 est dit au moins aussi préféré que le plan x_2 noté $x_1 \succsim x_2$ si et seulement si $U_{x_1} \geq U_{x_2}$.

Afin d'illustrer cette définition, l'utilité du plan 2 est calculée par rapport au modèle de préférences présenté sur la figure 3.5. Les utilités des différents critères du modèle ont été déterminées précédemment et valent respectivement $u^{t_2}_{\texttt{c-e1}} = 0.9$; $u^{t_2}_{\texttt{c-f1}} = 1$; $u^{t_2}_{\texttt{c-a1}} = 1$ et $u^{t_2}_{\texttt{c-s1}} = 0$. La capacité 2-additive μ utilisée est décrite sur la figure 3.5 par l'intermédiaire de ces coefficients de Möbius. Ainsi, l'utilité du plan 2 vaut $U = C_\mu(u^{t_2}_{\texttt{c-e1}}, u^{t_2}_{\texttt{c-f1}}, u^{t_2}_{\texttt{c-a1}}, u^{t_2}_{\texttt{c-s1}}) = 0.1 + 0.9 \times 0.25 + 0.15 + 0.2 = 0,675$.

L'extension du langage PDDL3 via l'exigence `maut-preferences` constitue l'une des contributions de cette étude. En effet, cette dernière enrichit le pouvoir expressif du langage PDDL3 en permettant d'utiliser facilement un nombre quelconque de préférences numériques, d'agréger des préférences élémentaires entre elles et de considérer d'éventuelles interactions entre les préférences du problème. De plus, une méthode outillée peut être utilisée pour construire efficacement le modèle de préférences correspondant. En conséquence, l'extension PDDL3/MAUT permet de représenter avec plus de précision les préférences des décideurs lors de la résolution des problèmes de planification.

Domaine de la gestion de crise

L'approche retenue pour faire cohabiter les préférences PDDL3 *avec le formalisme* MAUT *consiste à encoder les préférences* PDDL3 *au sein de critères préférentiels* MAUT. *Ceci est particulièrement intéressant pour le domaine de la gestion de crise puisque les décideurs peuvent ainsi utiliser à la fois des préférences* PDDL3 *et des préférences* MAUT *pour modéliser leurs attentes quant à la situation de crise qu'ils traitent. En plus des exemples mentionnés dans la section 1.2, ils peuvent ainsi représenter des préférences floues, des préférences non élémentaires ou encore prendre en compte les interactions entre leurs préférences. Une préférence non élémentaire pourrait par exemple être utilisée dans une situation où il serait souhaitable qu'une partie de la population soit la plus sédentaire possible (pour minimiser ses déplacements à la suite d'une inondation par exemple). Dans ce cas, des préférences relatives à la distribution de denrées alimentaires, à l'approvisionnement en électricité*

et à l'accès à l'information pourraient être combinées pour former un critère « d'incitation à la sédentarisation de la population » qu'il conviendrait de maximiser. Par ailleurs, dans le cas d'une inondation urbaine évaluée par rapport à deux critères relatifs à la surface de la ville inondée et à la hauteur d'eau maximale constatée, le modèle de préférences des décideurs pourrait inclure une interaction entre ces deux critères. Ainsi, un plan qui conduirait à une situation dans laquelle la surface inondée serait faible tandis que la hauteur d'eau serait grande pourrait être préféré à un plan qui conduirait à une situation dans laquelle la surface inondée et la hauteur d'eau seraient toutes deux moyennes.

3.2 Résolution du problème de planification avec préférences MAUT

3.2.1 Planification par recherche guidée par une heuristique

Les principales méthodes de l'art utilisées pour résoudre les problèmes de planification sont des techniques de *satisfaisabilité booléenne* [86, 87], de *satisfaction de contraintes* [41, 92, 108] ou de *recherche guidée par des heuristiques* [28, 80]. Les travaux présentés dans cette section s'appuient sur des mécanismes de recherche à l'aide d'heuristiques. Ces techniques sont les plus utilisées dans le domaine de la planification avec préférences notamment en raison de leur efficacité (cf. par exemple HPLAN-P [6] et LPRPG-P [39]).

Un graphe orienté $G = (V, E)$ est défini par la donnée d'un ensemble V de *nœuds* (ou *sommets*) et d'un ensemble E d'*arcs*. Les arcs sont des couples de sommets $e = (v_1, v_2)$ tels que $v_1, v_2 \in V$. Par ailleurs dans un graphe orienté, les arcs ont un sens ainsi (v_1, v_2) et (v_2, v_1) sont deux arcs différents. Un problème de planification peut être représenté par un graphe $G_P = (S, GA_\gamma)$ avec S l'ensemble des états dans lequel le système Σ étudié peut se trouver et GA_γ l'ensemble des transitions d'états représentées sous forme de paires de sommets. Par exemple, si $\gamma(s, a) = s'$ avec $a \in GA$ et $s, s' \in S$ alors l'arc $(s, s') \in GA_\gamma$. La résolution d'un problème de planification consiste donc à identifier un chemin de G_P qui relie l'état initial s_0 à

un état final $s \in S_{G'}$ avec $S_{G'} \subseteq S$ l'ensemble des états dans lesquels les objectifs G' sont vérifiés. Comme mentionné dans la section 1.1.2, les problèmes de planification sont généralement soumis à une telle combinatoire qu'il n'est pas envisageable de construire le graphe G_P entièrement. En conséquence, la construction de G_P doit être réalisée dynamiquement lors de la résolution du problème.

Plusieurs méthodes de recherche peuvent être employées pour résoudre un problème de planification. La *recherche en avant* part de l'état initial et essaye d'identifier un chemin qui mène à un état dans lequel les objectifs sont atteints. La *recherche en arrière* lui est diamétralement opposée puisqu'elle consiste à trouver un chemin des objectifs vers la situation initiale.

Les techniques de résolution proposées dans ce chapitre utilisent une méthode de recherche en avant à l'image de l'algorithme 3.1 présenté ci-après. La première étape de la recherche en avant consiste à construire l'ensemble des nœuds voisins du nœud initial. Le parcours du graphe se poursuit en sélectionnant l'un de ces nœuds qui est alors étendu à son tour de la même manière. Ainsi lors de la recherche, le graphe est naturellement divisé en deux parties : l'ensemble des *nœuds ayant été explorés* et l'ensemble des *nœuds non encore explorés* (cf. figure 3.6). La *frontière* est l'ensemble des nœuds non explorés qui peuvent être sélectionnés pour être étendus lors de la prochaine étape de l'algorithme. La notion de frontière est très importante puisque la façon dont cette dernière est étendue lors du parcours du graphe définit la *stratégie de recherche* employée.

La stratégie de recherche détermine le chemin à emprunter à partir de la frontière dans le but de progresser jusqu'à un nœud satisfaisant les objectifs. Il convient de distinguer les stratégies de recherche dites *aveugles* à l'image des stratégies de recherche en profondeur [55] ou en largeur [96] des stratégies dites *informées* à l'image de la recherche par le *meilleur en premier* (ou encore *Best-First-Search* ou *BFS*) [113]. Dans le cas des stratégies de recherche aveugles, aucune information spécifique au problème à résoudre n'est utilisée pour choisir les nœuds de la frontière qui sont étendus. Par exemple, avec une stratégie de recherche en profondeur (respectivement en largeur), le nœud de la frontière sélectionné pour être étendu est celui dont la profondeur (nombre d'arcs qui le sépare du nœud initial) est la plus grande (respectivement la plus petite). Ainsi, en considérant la frontière présentée sur la figure 3.6, le nœud A (respectivement C) serait étendu par une

stratégie de recherche de type en profondeur (respectivement en largeur). En revanche, les stratégies informées sont caractérisées par le fait qu'elles s'appuient sur des informations spécifiques au problème à résoudre. Par exemple, dans le cadre d'une stratégie BFS, les nœuds de la frontière sont triées en fonction de leur potentiel ; lequel est évalué à l'aide d'une heuristique. Le meilleur nœud au sens de l'heuristique considérée est alors retenu pour poursuivre la recherche. Si la fonction heuristique f doit être minimisée, le nœud E de la frontière de la figure 3.6 serait alors sélectionné par la stratégie BFS.

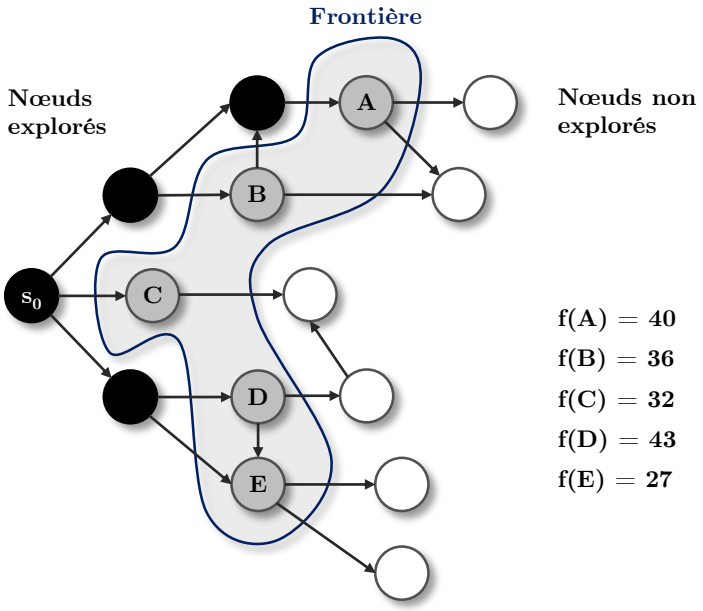


FIGURE 3.6 – Principe de la recherche en avant dans un graphe

L'algorithme 3.1 est un algorithme générique de résolution de problèmes de planification par recherche en avant dans un graphe. Il s'agit d'un algorithme itératif qui cherche des solutions de qualité croissante et ce jusqu'à ce que les ressources à disposition (temps alloué au calcul notamment) soient épuisées ce qui est caractérisé par la fonction `AreResourcesExhausted()`. Pour instancier l'algorithme 3.1, il faut préciser une *règle de sélection* ainsi qu'un ensemble (éventuellement vide) de *règles de coupe*. La règle de sélection implémente la stratégie de recherche et les mécanismes itératifs de l'algorithme. Les règles de coupe sont quant à elles utilisées pour identifier et supprimer les nœuds qu'il n'est pas nécessaire de visiter lors de la suite de la recherche.

L'algorithme consiste à sélectionner (`selectionRule.selectNode()`) puis traiter des nœuds de la frontière. Si le nœud sélectionné satisfait les objectifs à atteindre, une solution a alors été identifiée et l'algorithme débute une nouvelle itération (`selectionRule.nextIteration()`). De plus, si la qualité de cette solution est supérieure à la meilleure solution connue, cette dernière est ajoutée à la liste des solutions à retourner lorsque l'algorithme se termine. En revanche, si le nœud sélectionné ne satisfait pas les objectifs à atteindre, l'algorithme se poursuit en construisant son voisinage à partir de l'ensemble des actions du problème. Si une action est applicable (`IsApplicable()` $= true$) dans le nœud considéré et qu'elle est admissible par rapport aux règles de coupe (`IsPrunable()` $= false$), le nœud est alors étendu (`ExpandNode()`) conformément à la sémantique présentée dans les sections 1.1.3, 1.2.3 et 3.1.3. Dans ce cas, la frontière de la recherche est alors mise à jour (`selectionRule.updateFrontier()`) en conséquence.

Des règles de sélection pour la planification avec préférences sont proposées dans la section 3.2.2. Par ailleurs, les règles d'applicabilité (qui définissent le comportement de la fonction `IsApplicable()`) sont présentées avec les règles de coupe dans la section 3.2.3 puisque leurs implémentations respectives sont similaires. Les règles d'applicabilité ne diffèrent des règles de coupe que par le fait que leur présence est obligatoire dans l'algorithme pour garantir la validité des plans solutions retournés. En effet, pour un état s et une action a, ces règles autorisent l'extension de s par a uniquement si les préconditions de a sont vérifiées dans s et si l'exécution de a dans s ne viole aucun des objectifs de trajectoire du problème.

Input :	param paramètres qui précisent les ressources disponibles
	s_0 état initial du problème
	G formule des objectifs à atteindre
	GA ensemble des actions closes du problème
Output :	solution liste de plans solutions
Data :	selectionRule la règle de sélection utilisée
	prunningRules ensemble des règles de coupe utilisées
	current nœud courant de l'algorithme
	child nœud obtenu par extension du nœud courant

```
Algorithm SEARCH(param, s_0, G, GA)
    selectionRule.initFrontier(s_0)
    while ARERESOURCESEXHAUSTED(param) = false and
    selectionRule.isFrontierEmpty() = false do
        current ← selectionRule.selectNode()
        if current.evaluateFormula(G) = true then
            if current.getUtility() > solution.getBestUtility() then
                solution.add(current)
            end
            selectionRule.nextIteration(s_0)
        else
            for all action ∈ GA do
                if ISAPPLICABLE(current, action) = true and
                 ISPRUNABLE(prunningRules, current, action) = false then
                    child ← EXPANDNODE(current, action)
                    selectionRule.updateFrontier(child)
                end
            end
        end
    end
    return solution

Function ISPRUNABLE(prunningRules, current, action)
    for all prunningRule ∈ prunningRules do
        if prunningRule.IsActionPrunable(current, action) = true then
            return true
        end
    end
    return false
```

Algorithme 3.1 : Résolution itérative de problèmes de planification par recherche dans un graphe. Les objets `selectionRule` et `pruningRules` doivent être précisés pour décrire complètement la méthode de résolution.

3.2.2 Règles de sélection pour la planification avec préférences

La règle de sélection est l'élément clef de l'algorithme 3.1 puisqu'elle détermine les performances de ce dernier. Pour définir une règle de sélection, il faut préciser une *stratégie de recherche* et une *heuristique*. La stratégie de recherche caractérise le comportement de la règle de sélection lorsqu'une solution est identifiée (via la fonction `nextIteration()`) et définit les mécanismes d'extension de la frontière lors de la recherche (cf. fonctions `initFrontier()`, `isFrontierEmpty()`, `selectNode()` et `updateFrontier()`). Ces mécanismes s'appuient sur une heuristique $h(I_M, s)$ (fonction `evaluateHeuristic()`) qui estime le potentiel d'un nœud s par rapport aux objectifs et aux préférences de l'instance de planification I_M considérée. Dans le cas de la planification avec préférences, les heuristiques peuvent être relativement complexes à écrire puisqu'elles doivent réaliser un compromis entre objectifs et préférences. En effet, si la recherche est focalisée sur les objectifs, il est alors facile d'identifier des solutions mais ces dernières ont de fortes chances d'être médiocres au regard du modèle de préférences considéré. En revanche, si la recherche est seulement focalisée sur les préférences, il se peut qu'aucune solution ne soit jamais identifiée en raison de la forte combinatoire des problèmes de planification. La solution retenue dans cette étude consiste à construire les heuristiques $h(I_m, s)$ à partir d'estimations $\Delta(I_M, s)$ et $\Lambda(I_M, s)$ qui jugent respectivement le nœud s seulement par rapport aux objectifs du problème et seulement par rapport au modèle de préférences considéré.

Ces travaux proposent six règles de sélection dont les performances sont évaluées dans la section 3.3. Ces dernières respectent toutes la structure générique des règles de sélection qui est précisée par l'algorithme 3.2. Afin de faciliter la compréhension du lecteur, leur construction est exposée selon une approche bottom-up. Pour commencer, trois estimations Δ_1, Λ_1 et Λ_2 sont décrites. Trois heuristiques h_1, h_2 et h_3 sont ensuite élaborées par agrégation de ces estimations. Finalement, deux stratégies de recherche SR_1 et SR_2 sont présentées. SR_1 et SR_2 peuvent chacune être utilisées conjointement avec h_1, h_2 ou h_3.

Object SELECTIONRULE

- **Function** `nextIteration()`
- **Function** `initFrontier(`s_0`)`
- **Function** `isFrontierEmpty()`
- **Function** `selectNode()`
- **Function** `updateFrontier(node)`
- **Function** `evaluateHeuristic(node)`

Algorithme 3.2 : Structure d'une règle de sélection

Distance d'un nœud aux objectifs

Cette section s'intéresse à $\Delta_1(I_M, s)$ qui constitue une estimation du nombre minimal d'actions qu'il faut exécuter pour atteindre les objectifs G' de I_M à partir du nœud s. Un nœud est d'autant plus intéressant que sa distance aux objectifs est faible. En effet, privilégier de tels nœuds lors de la recherche permet généralement d'identifier rapidement une solution au problème. Le calcul de Δ_1 repose sur l'heuristique employée par le planificateur FAST-FORWARD [80,81]. Pour que le calcul puisse être réalisé efficacement (en temps polynomial par rapport aux données d'entrées du problème), cette heuristique utilise une version simplifiée I'_M du problème à résoudre. Ainsi, le problème est *relaxé* en ignorant les effets négatifs des actions (méthode déjà employée auparavant dans [27,28]) et en ignorant les objectifs étendus du problème (la notion d'objectifs étendus étant en réalité postérieure au planificateur FAST-FORWARD). L'heuristique proposée par FAST-FORWARD consiste à appliquer la procédure de résolution par *graphe de planification* proposée dans GRAPHPLAN [25,26] avec le problème de planification relaxé.

Afin d'expliquer le calcul de $\Delta_1(I_M, s)$, la notion de *graphe de planification relaxé* est précisée. Il s'agit d'une structure composée de plusieurs ensembles de prédicats P_i et de plusieurs ensembles d'actions A_i qui se succèdent les uns aux autres (voir tableau 3.1 et figure 3.7 pour un exemple). Le premier ensemble de prédicats P_0 contient l'ensemble des atomes de s. Le premier ensemble d'action A_1 est quant à lui obtenu en sélectionnant l'ensemble des actions du problème applicables dans

P_0. Le deuxième ensemble de prédicats P_1 contient tous les prédicats de P_0 (ce qui s'interprète comme l'utilisation d'un opérateur fictif `No-Op` sur chacun de ces prédicats) ainsi que l'ensemble des effets *positifs* des actions de A_1. La construction du graphe de planification relaxé se poursuit de la sorte jusqu'à ce qu'un ensemble de prédicats P_m contenant tous les objectifs à atteindre soit construit.

Une fois que le graphe de planification relaxé a été élaboré, il est possible d'en extraire une solution (voir [79] pour l'algorithme exact). La solution est une séquence d'ensembles d'actions $\langle O_1, \cdots, O_m \rangle$ qui est obtenue par recherche en arrière dans le graphe de planification relaxé. Ainsi, pour construire l'ensemble O_m, il faut commencer par identifier les prédicats p_j^m de P_m qui correspondent aux objectifs à atteindre. Si un prédicat p_j^m ne peut pas être obtenu par l'opérateur `No-Op` et si l'action a qui permet de l'obtenir n'appartient pas à un ensemble d'actions A_i tel que $i < m$ (les actions ne vérifiant pas cette condition n'ont pas été représentées sur la figure 3.7 afin de simplifier l'illustration), alors l'action a est ajoutée à l'ensemble d'actions O_m. Les prédicats p_j^m correspondant aux objectifs à atteindre pouvant être obtenus par l'opérateur `No-Op` et les préconditions des actions de O_m deviennent alors les prédicats p_j^{m-1} à considérer pour construire l'ensemble d'actions O_{m-1}. L'algorithme d'extraction de la solution se termine après que l'ensemble O_1 ait été construit. Dans l'exemple proposé par le tableau 3.1 et la figure 3.7, les prédicats p_j^3 de P_3 à considérer pour construire O_3 sont p_3 et p_5. Le prédicat p_3 peut être obtenu par l'opérateur `No-Op` (représenté par une flèche en pointillés sur la figure 3.7) tandis que le prédicat p_5 peut être obtenu par l'action a_4 qui constitue donc l'unique action de O_3. La solution extraite du graphe de planification relaxé de cet exemple est $\langle \{a_1\}, \{a_2\,;\, a_3\}, \{a_4\} \rangle$.

Action	**Préconditions**	**Effets Positifs**	**Effets Négatifs**
1	p_1	p_2	$\neg\, p_3$
2	p_2	p_3	-
3	p_2	p_4	$\neg\, p_1$
4	p_1, p_4	p_5	$\neg\, p_4$
5	p_4	p_3	-

TABLEAU 3.1 – Actions pour illustrer la notion de graphe de planification relaxé

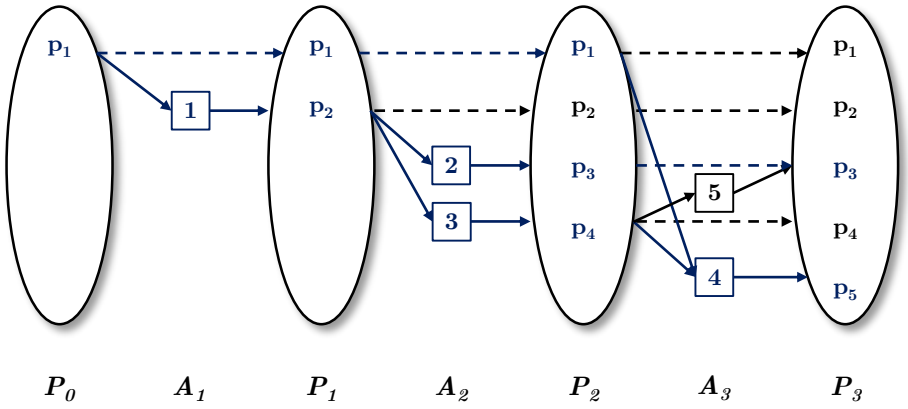


FIGURE 3.7 – Graphe de planification relaxé de $s = \{p_1\}$ à $G = \{p_3, p_5\}$. En bleu, l'illustration de la méthode de construction de la solution.

La valeur de Δ_1 est identique à celle de l'heuristique proposée par FAST-FORWARD et correspond à la taille de la solution $\langle O_1, \cdots, O_m \rangle$ extraite du graphe de planification relaxé (cf. définition 3.6). Ainsi dans l'exemple proposé, la distance du nœud s aux objectifs G vaut $\Delta_1(I_M, s) = 4$.

Définition 3.6 - Estimation Δ_1
Soit I_M une instance d'un problème de planification et s un état. Soit $\texttt{FF}(I_M, s, G)$ l'heuristique employée par FAST-FORWARD et $\langle O_1, \cdots, O_m \rangle$ la solution extraite du graphe de planification relaxé de s vers les objectifs G de I'_M. L'estimation $\Delta_1(I_M, s)$ est définie par :

$$\Delta_1(I_M, s) \; = \; \texttt{FF}(I_M, s, G) \; = \; \sum_{i=1}^{m} \; |O_i|$$

Qualité d'un nœud par rapport aux préférences

Cette section s'intéresse aux estimations $\Lambda_1(I_M, s)$ et $\Lambda_2(I_M, s)$ qui jugent le potentiel d'un nœud s au regard des préférences du problème. Les méthodes à mettre en œuvre pour calculer ces valeurs diffèrent nécessairement de celles employées dans le cas de Δ_1 en raison de la différence sémantique qu'il existe entre objectifs et préférences. En effet, puisque tous les objectifs d'un problème doivent être vérifiés dans un plan pour qu'il soit valide, ces derniers sont tous aussi importants les uns que les autres. En revanche, les préférences n'impactent généralement pas

le calcul de l'utilité du plan de façon homogène (sauf dans le cas de modèles de préférences particuliers). Il en résulte naturellement que deux plans satisfaisant des sous-ensembles de préférences différents ne sont pas évalués de manière identique. Cette distinction sémantique entre objectifs et préférences, aussi évidente soit elle, met en exergue la nécessité de considérer l'*importance* individuelle de chaque préférence du modèle pour élaborer les estimations $\Lambda_1(I_M, s)$ et $\Lambda_2(I_M, s)$.

La grandeur $\Lambda_1(I_M, s)$ constitue une estimation de l'utilité finale de tout plan qui serait construit par extension de la trajectoire $\langle s_0, \cdots, s\rangle$. Elle consiste simplement à évaluer la fonction objectif dans le nœud s (voir définition 3.7). Un tel calcul est rapide à réaliser et tient naturellement compte de l'importance relative des différentes préférences du problème. En s'appuyant sur la valeur d'utilité *locale* observée dans le nœud s, cette estimation est optimiste vis à vis des préférences satisfaites dans s et pessimiste vis à vis des préférences violées dans s.

Définition 3.7 - Estimation Λ_1
Soit I_M une instance d'un problème de planification et t la trajectoire de l'état initial s_0 à l'état s. Si $u_1^t, \cdots, u_p^t$ sont les valeurs d'utilités partielles des critères MC de I_M par rapport à la trajectoire $\langle s_0, \cdots, s\rangle$, alors :

$$\Lambda_1(I_M, s) \;=\; C_\mu\left(u_1^t, \cdots, u_p^t\right)$$

L'estimation Λ_1 tient compte de l'importance relative des différentes préférences du problème mais ne considère aucune information quant à l'évolution future de la satisfaisabilité de ces dernières. Il est possible de proposer une estimation qui combine pour chaque préférence la *difficulté* inhérente à sa satisfaction avec son *importance*. A titre d'exemple, l'une des heuristiques proposées dans [6] consiste à évaluer la fonction objectif du problème dans les différents nœuds P_i d'un graphe de planification relaxé puis à pondérer ces valeurs en fonction de la profondeur i des nœuds P_i considérés. Par ailleurs, l'heuristique utilisée dans LPRPG-P [39] peut être interprétée comme une modification de la structure de graphe de planification relaxé qui permet de tenir compte des préférences du problème et de leurs poids respectifs. L'approche retenue dans cette étude utilise également le concept de graphe de planification relaxé mais repose sur la nature floue des critères MAUT utilisés pour représenter les préférences du problème.

La valeur des préférences finales et des préférences de trajectoire est par nature binaire puisque ces dernières ne peuvent se trouver que dans deux états (satisfaites ou violées). La grandeur $\Lambda_2(I_M, s)$ est construite en remplaçant l'interprétation usuelle de ces préférences par une estimation de l'effort à produire pour les satisfaire à partir du nœud s. Il convient de remarquer que cette interprétation est compatible avec la sémantique initiale de ces préférences puisque les deux cas limites d'un effort nul ou d'un effort infini s'interprètent respectivement comme le fait que la préférence soit satisfaite ou violée dans s.

Le calcul de $\Lambda_2(I_M, s)$ est analogue à celui de $\Lambda_1(I_M, s)$ puisqu'il consiste à évaluer la fonction objectif du problème par rapport à la trajectoire $\langle s_0, \cdots, s \rangle$. Néanmoins, dans le cas de $\Lambda_2(I_M, s)$, le calcul des valeurs d'utilités partielles des préférences diffère. En effet, en raison de la modification précédemment mentionnée, les critères MAUT associées aux préférences sont redéfinis. Dans le cas d'une préférence finale p représentée par un critère k, la valeur y_k de ce critère est déterminée à partir de l'heuristique FF. En effet, la grandeur $\texttt{FF}(I_M, s, p)$ constitue bien une estimation de l'effort à fournir pour satisfaire p depuis le nœud s. Néanmoins, il est impossible d'associer à cette estimation une fonction d'utilité partielle qui serait valable pour tout problème de planification. En effet, il se peut que $\texttt{FF}(I_M^1, s_1, p_1) = \texttt{FF}(I_M^2, s_2, p_2) = 15$ en considérant que s_1 est un nœud à fort potentiel au vue de la complexité de l'instance I_M^1 mais que s_2 est un nœud à faible potentiel au vue de la complexité de l'instance I_M^2. Afin de résoudre ce problème, la valeur y_k est définie comme le rapport entre $\texttt{FF}(I_M, s, p)$ et $\texttt{FF}(I_M, s_0, p)$. La valeur du critère s'interprète donc comme l'effort à fournir pour satisfaire la préférence p depuis le nœud s par rapport à l'effort à fournir pour la satisfaire depuis le nœud représentant l'état initial. Il devient alors possible de caractériser entièrement le critère k associé à une préférence finale quelconque p et ce quelque soit le problème de planification considéré. L'espace de définition de k est $\Omega_k = \mathbb{R}^+$ avec $\mathbf{1}_k = 0$ et $\mathbf{0}_k = 5$ les éléments parfaitement satisfaisant et totalement insatisfaisant de k. La fonction d'utilité partielle u_h du critère k est représentée sur la figure 3.8. Lorsque le rapport y_k vaut 1, la satisfaction associée à la préférence est moyenne (l'effort à fournir pour atteindre p depuis s est le même que celui pour l'atteindre depuis s_0). Cette dernière augmente lorsque la valeur de y_k diminue (l'effort à fournir pour atteindre p depuis s est plus faible que celui pour l'atteindre depuis s_0). En revanche, la satisfaction associée à la préférence diminue lorsque la valeur

y_k augmente (l'effort à fournir pour atteindre p depuis s est plus grand que celui pour l'atteindre depuis s_0). Pour finir, la définition 3.8 explicite formellement le calcul de la valeur d'utilité partielle associée à une préférence finale dans le cadre de l'estimation Λ_2.

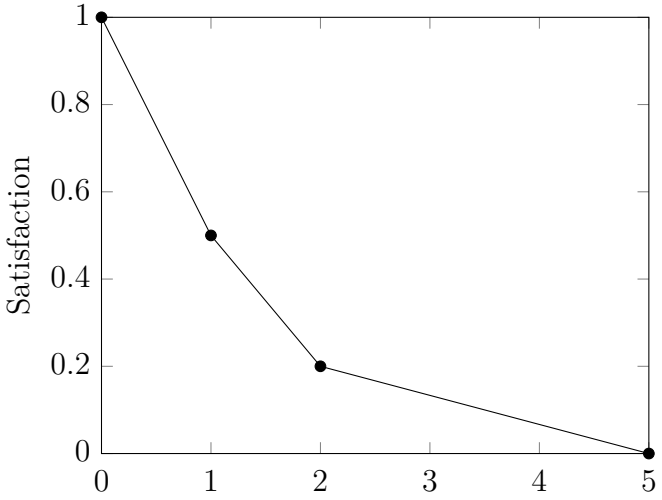


FIGURE 3.8 – Fonction d'utilité partielle u_h

Définition 3.8 - Utilité d'une préférence finale dans Λ_2
Soit I_M une instance d'un problème de planification, p la préférence relative à l'objectif final ϕ représentée par le critère k, s un état et t la trajectoire $\langle s_0, \cdots, s \rangle$. Soit $u_h : [0, 5] \rightarrow [0, 1]$ la fonction d'utilité partielle linéaire par morceaux définie telle que $u_h(0) = 1$; $u_h(1) = 0.5$, $u_h(2) = 0.2$ et $u_h(5) = 0$. La valeur d'utilité partielle du critère associé à la préférence finale ϕ est définie par :

$$u_k^t = \begin{cases} u_h\left(\dfrac{\texttt{FF}(I_M, s, \phi)}{\texttt{FF}(I_M, s_0, \phi)}\right) \text{ si } \texttt{FF}(I_M, s_0, \phi) \neq 0\,; \\ 1 \text{ si } \texttt{FF}(I_M, s_0, \phi) = 0 \text{ et } \texttt{FF}(I_M, s, \phi) = 0\,; \\ 0 \text{ si } \texttt{FF}(I_M, s_0, \phi) = 0 \text{ et } \texttt{FF}(I_M, s, \phi) \neq 0\,; \end{cases}$$

Les deux dernières conditions de la définition 3.8 permettent de s'assurer que la valeur u_k^t est correctement définie même lorsque $\texttt{FF}(I_M, s_0, \phi) = 0$ (c.-à-d. quand la préférence p est satisfaite dans l'état initial s_0 du problème). Dans ce cas, la valeur d'utilité u_k^t vaut 1 si p est également satisfaite dans s et 0 si elle ne l'est pas.

Les préférences de trajectoires sont plus complexes à traiter puisque leur sémantique est relativement riche. Cette étude ne s'intéresse qu'aux préférences de trajectoires non temporelles du PDDL3 à savoir `always`, `sometime`, `at-most-once` et `sometime-before` (voir définition 1.23). Le traitement de ces quatre types de préférences dans le cadre de l'estimation Λ_2 est à présent précisé. Le cas des préférences `always` et `at-most-once` est relativement simple puisqu'aucune modification n'est apportée au calcul de la définition 3.3. En effet, leur sémantique se prête peu à l'introduction d'une notion de difficulté ou d'effort à fournir pour satisfaire la préférence depuis un nœud du problème. Par exemple, considérer l'effort à produire pour satisfaire le prédicat ϕ d'une préférence `always` ϕ depuis un nœud s n'a aucun sens puisque si ϕ n'est pas vraie dans s alors la préférence est violée. De même, pour une préférence `at-most-once` ϕ, chercher à satisfaire ϕ ou $\neg\,\phi$ peut avoir des effets contre-productifs au vue de la sémantique de cette préférence. En revanche, la sémantique associée aux préférences de trajectoire `sometime` et `sometime-before` peut être modifiée avantageusement dans le cadre du calcul de Λ_2. Pour expliciter ces modifications, les automates de Büchi [33, 133] de ces préférences sont représentés sur la figure 3.9 [47, 65]. Ces derniers précisent l'évolution de la valeur de vérité de ces préférences lorsqu'une transition d'états survient. Par exemple, la préférence `sometime` ϕ est initialement dans l'état S_0 ; lequel est un état de la préférence et ne doit pas être confondu avec s_0 qui lui est un état du problème. Si une action ayant pour effet positif ϕ (respectivement effet négatif $\neg\,\phi$) est exécutée dans s_0, le problème évolue vers un état s dans lequel l'état de la préférence est S_2 (respectivement S_1).

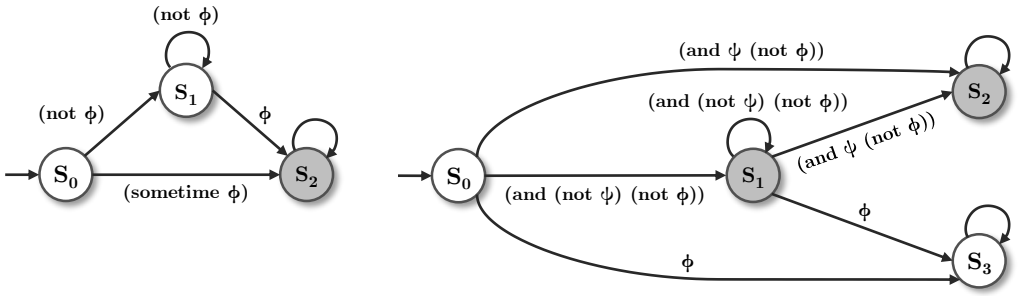


FIGURE 3.9 – Automates de Büchi des préférences `sometime` ϕ et `sometime-before` $\phi\ \psi$; d'après [65]

L'automate de Büchi de `sometime` ϕ ne possède qu'un seul état dans lequel la préférence est satisfaite (cercle gris sur la figure 3.9). Cet état S_2 n'est accessible que lorsque ϕ devient vraie ce qui conduit naturellement à caractériser l'effort à fournir pour satisfaire cette préférence depuis un nœud s en fonction du prédicat ϕ (cf. définition 3.9). Au vu des mécanismes utilisés, le calcul de l'utilité d'une préférence `sometime` pour l'estimation Λ_2 est analogue à celui présenté dans la définition 3.8.

Définition 3.9 - Utilité d'une préférence `sometime` dans Λ_2
Soit I_M une instance d'un problème de planification, p une préférence de trajectoire de type `sometime` ϕ représentée par le critère k, s un état et t la trajectoire $\langle s_0, \cdots, s \rangle$. Soit $u_h : [0,5] \rightarrow [0,1]$ la fonction d'utilité partielle linéaire par morceaux définie telle que $u_h(0) = 1$; $u_h(1) = 0.5$, $u_h(2) = 0.2$ et $u_h(5) = 0$. La valeur d'utilité partielle du critère k est définie par :

$$u_k^t \;=\; \begin{cases} u_h\left(\dfrac{\texttt{FF}(I_M, s, \phi)}{\texttt{FF}(I_M, s_0, \phi)} \right) \;\text{ si }\; \texttt{FF}(I_M, s_0, \phi) \neq 0\,; \\ 1 \;\text{ si }\; \texttt{FF}(I_M, s_0, \phi) = 0 \;\text{ et }\; \texttt{FF}(I_M, s, \phi) = 0\,; \\ 0 \;\text{ si }\; \texttt{FF}(I_M, s_0, \phi) = 0 \;\text{ et }\; \texttt{FF}(I_M, s, \phi) \neq 0\,; \end{cases}$$

L'automate de Büchi de `sometime-before` ϕ ψ est plus complexe notamment parce qu'il possède deux états (S_1 et S_2) dans lesquels la préférence est satisfaite. L'état S_2 est le plus intéressant puisque la préférence ne peut plus jamais être violée lorsqu'il est atteint. Pour qu'une transition d'états conduise à S_2, cette dernière doit générer un état dans lequel ψ et $\neg\,\phi$ sont simultanément vrais. La condition $\neg\,\phi$ est problématique puisque l'heuristique `FF` s'appuie sur le concept de graphe de planification relaxé qui par définition ne considère pas les effets négatifs du problème. En conséquence, le calcul de la valeur d'utilité partielle associée à une préférence de type `sometime-before` ϕ ψ est réalisé en fonction du prédicat ψ. Ce dernier tient néanmoins compte de la valeur de vérité de `sometime-before` ϕ ψ (via la condition $t \models p$ dans la définition 3.10) afin de respecter la sémantique initiale de la préférence.

Définition 3.10 - Utilité d'une préférence `sometime-before` dans Λ_2
Soit I_M une instance d'un problème de planification, p une préférence de trajectoire de type `sometime-before` ϕ ψ représentée par le critère k, s un état et t la trajectoire $\langle s_0, \cdots, s \rangle$. Soit $u_h : [0,5] \rightarrow [0,1]$ la fonction d'utilité partielle linéaire par morceaux définie telle que $u_h(0) = 1$; $u_h(1) = 0.5$, $u_h(2) = 0.2$ et $u_h(5) = 0$. La valeur d'utilité partielle de k est définie par :

$$u_k^t \;=\; \begin{cases} u_h\left(\dfrac{\texttt{FF}(I_M, s, \phi)}{\texttt{FF}(I_M, s_0, \phi)}\right) \;\text{ si }\; t \models p \;\text{ et }\; \texttt{FF}(I_M, s_0, \phi) \neq 0\,; \\ 1 \;\text{ si }\; t \models p \;\text{ et }\; \texttt{FF}(I_M, s_0, \phi) = 0 \;\text{ et }\; \texttt{FF}(I_M, s, \phi) = 0\,; \\ 0 \;\text{ si }\; t \models p \;\text{ et }\; \texttt{FF}(I_M, s_0, \phi) = 0 \;\text{ et }\; \texttt{FF}(I_M, s, \phi) \neq 0\,; \\ 0 \;\text{ si }\; t \models \neg p\,; \end{cases}$$

Les préférences numériques ne sont pas modifiées dans le cadre du calcul de Λ_2 puisque leur nature floue est par définition pleinement exploitée. Une extension envisageable consisterait à tenir compte de l'évolution potentielle des préférences numériques entre le nœud s et les nœuds objectifs. Pour conclure, le calcul de l'estimation Λ_2 est précisé par la définition 3.11.

Définition 3.11 - Estimation Λ_2
Soit I_M une instance d'un problème de planification, s un état et t la trajectoire $\langle s_0, \cdots, s \rangle$. Soit $u_1^t, \cdots, u_p^t$ les valeurs d'utilités partielles des critères de I_M calculés par rapport à t en utilisant les définitions 3.8, 3.9 et 3.10 en priorité plutôt que la définition 3.3. L'estimation Λ_2 est définie par :

$$\Lambda_2(I_M, s) \;=\; C_\mu\left(u_1^t, \cdots, u_p^t\right)$$

Heuristiques pour la planification avec préférences

Les trois estimations Δ_1, Λ_1 et Λ_2 ayant été présentées, les heuristiques h_1, h_2 et h_3 peuvent à présent être introduites. Le fonctionnement de ces trois heuristiques est relativement similaire. En effet, elles privilégient toutes les nœuds ayant un fort potentiel vis à vis des objectifs à atteindre (caractérisé par Δ_1) tout en tenant

compte du potentiel de ces derniers par rapport aux préférences du problème (caractérisé par Λ_1 et Λ_2). Une approche similaire est utilisée par le planificateur HPLAN-P dont l'heuristique est élaborée à partir de plusieurs estimations du potentiel d'un nœud ; lesquelles sont agrégées via un ordre lexicographique [6]. En effet, les nœuds sont dans un premier temps triés en fonction de leur distance par rapport aux objectifs puis les éventuelles égalités sont départagées à l'aide d'une fonction qui estime le potentiel des nœuds par rapport aux préférences du problème. L'approche retenue dans cette étude consiste à construire les heuristiques h_1, h_2 et h_3 à partir de trois agrégations différentes des grandeurs Δ_1, Λ_1 et Λ_2.

Pour réaliser ces agrégations, il convient dans un premier temps de rendre les grandeurs Δ_1, Λ_1 et Λ_2 commensurables. En tant que score calculé sur la base d'un modèle de préférences MAUT, Λ_1 et Λ_2 sont toutes deux à valeur dans $\xi = [0, 1]$ et représentent une satisfaction quant à la qualité potentielle d'un nœud s. Ce n'est en revanche pas le cas de l'estimation Δ_1. Construite à partir de Δ_1 en utilisant les mécanismes précédemment décrits et la fonction d'utilité partielle u_h (cf. figure 3.8), la grandeur Δ'_1 est quant à elle commensurable avec Λ_1 et Λ_2.

Définition 3.12 - Estimation Δ'_1
Soit I_M une instance d'un problème de planification, s un état et $u_h : [0, 5] \rightarrow [0, 1]$ la fonction d'utilité partielle linéaire par morceaux définie telle que $u_h(0) = 1$; $u_h(1) = 0.5$; $u_h(2) = 0.2$ et $u_h(5) = 0$. $\Delta'_1(I_m, s)$ est définie par :

$$\Delta'_1(I_M, s) \;=\; u_h\left(\frac{\Delta_1(s, G)}{\texttt{FF}(I_M, s_0, G)}\right)$$

L'opérateur d'agrégation retenu pour construire les heuristiques h_1, h_2 et h_3 est une intégrale de Choquet 2-additive de capacité ρ. Il convient de ne pas confondre la fonction de capacité ρ qui spécifie l'agrégation réalisée par les heuristiques h_1, h_2 et h_3 avec la fonction de capacité μ qui est utilisée pour agréger les différentes préférences du problème à résoudre. L'expression de ρ est explicitée dans la section suivante lorsque les aspects itératifs de l'algorithme sont précisés.

Définition 3.13 - Heuristique h_1
Soit I_M une instance d'un problème de planification, s un état et C_ρ une intégrale de Choquet 2-additive de capacité ρ. La valeur de l'heuristique h_1 dans le nœud s est définie par :

$$h_1(I_M, s) \;=\; C_\rho\Big(\,\Delta'_1(I_M, s)\,,\,\Lambda_1(I_M, s)\,\Big)$$

Définition 3.14 - Heuristique h_2
Soit I_M une instance d'un problème de planification, s un état et C_ρ une intégrale de Choquet 2-additive de capacité ρ. La valeur de l'heuristique h_2 dans le nœud s est définie par :

$$h_2(I_M, s) \;=\; C_\rho\Big(\,\Delta'_1(I_M, s)\,,\,\Lambda_2(I_M, s)\,\Big)$$

Définition 3.15 - Heuristique h_3
Soit I_M une instance d'un problème de planification, s un état et C_ρ une intégrale de Choquet 2-additive de capacité ρ. Soit $\Lambda_3(I_M, s) = 0,6 * \Lambda_2(I_M, s) + 0,4 * \Lambda_1(I_M, s)$. La valeur de l'heuristique h_3 dans le nœud s est définie par :

$$h_3(I_M, s) \;=\; C_\rho\Big(\,\Delta'_1(I_M, s)\,,\,\Lambda_3(I_M, s)\,\Big)$$

En ce qui concerne la définition 3.15, Λ_3 a été construit à partir d'une pondération entre Λ_1 et Λ_2 afin que h_3 soit similaire à h_1 et h_2. Ceci permet notamment aux trois heuristiques d'utiliser la même fonction de capacité ρ. Néanmoins, l'heuristique h_3 aurait également pu être définie par $h_3(I_M, s) = C_{\rho'}(\Delta'_1(I_M, s), \Lambda_1(I_M, s), \Lambda_2(I_M, s))$; auquel cas une seconde fonction de capacité ρ' aurait du être spécifiée.

Pour généraliser, il convient de préciser qu'un nombre quelconque d'estimations pourraient être utilisées pour construire ce type d'heuristiques. Intuitivement, plus le nombre de critères d'estimation utilisé augmente et plus la probabilité de faire face à des plateaux (un grand nombre de solutions ayant le même score) au cours de la recherche diminue. En revanche, il devient alors plus difficile de construire la fonction de capacité ρ.

Stratégies de recherche

Trois heuristiques h_1, h_2 et h_3 ont été proposées pour implémenter la fonction `EvaluateHeuristic()` de la règle de sélection. Cette section s'intéresse quant à elle à la définition des fonctions `initFrontier()`, `isFrontierEmpty()`, `selectNode()`, `updateFrontier()` et `nextIteration()` qui caractérisent la stratégie de recherche de la règle de sélection (cf. algorithmes 3.1 et 3.2).

Les deux stratégies de recherche SR_1 et SR_2 considérées dans cette étude utilisent une stratégie de recherche de type le meilleur nœud en premier (BFS). Celle-ci est implémentée à l'aide de deux listes afin d'éviter d'éventuels problèmes de boucles infinies lors de la recherche (cf. algorithme 3.3). La liste `closeList` contient l'ensemble des nœuds déjà explorés lors du parcours du graphe. L'objet `openList` est quant à lui une *liste ordonnée* qui représente la frontière de la recherche. Ainsi, la position d'un nœud dans `openList` est déterminée par sa valeur de coût (`cost`) ; laquelle est calculée via l'heuristique (`EVALUATEHEURISTIC()`) de la règle de sélection. La position d'un nœud dans `openList` est déterminée automatiquement lors de l'ajout du nœud à la liste ou lors de l'appel à la fonction `sort()`.

Input : s_0 état initial du problème
child nœud de l'espace de recherche

Data : closeList liste des nœuds explorés
openList liste ordonnée des nœuds à explorer

Object SELECTIONRULE

Function `initFrontier(`s_0`)`

```
openList ← newSortedList()
closeList ← newList()
openList.add(s0)
```

```
Function isFrontierEmpty()
    return ( openList.size() = 0 )

Function selectNode()
    node ← openList.getFirstElement()
    openList.remove(node)
    closeList.add(node)
    return node

Function updateFrontier(child)
    cost ← EvaluateHeuristic(child)
    if openList.contains(child) = false and
    closeList.contains(child) = false then
        child.setCost(cost)
        openList.add(child)
    else if child.getCost() > cost then
        child.setCost(cost)
        if openList.contains(child) = false then
            openList.add(child)
        else
            openList.sort()
        end
    end
```

Algorithme 3.3 : Implémentation d'une règle de sélection de type BFS. La fonction `EvaluateHeuristic()` détermine la valeur d'un nœud au regard de l'heuristique considérée. Les fonctions `newList()` et `newSortedList()` retournent respectivement une structure de données de type liste et de type liste ordonnée. Les fonctions `add()`, `getFirstElement()`, `size()` et `contains()` s'interprètent comme les opérateurs usuels d'une liste.

L'algorithme 3.1 est un algorithme itératif puisqu'il retourne une succession de plans de qualité croissante au cours de la recherche (voir définition 1.29, section 1.2.4). Cette propriété est très appréciable dans le cadre de la planification avec préférences puisque la complexité combinatoire des problèmes rend généralement impossible l'identification d'une solution optimale. Les modifications apportées à la stratégie de recherche lorsqu'une nouvelle solution est identifiée sont définies dans la fonction `nextIteration()` de la règle de sélection.

L'approche retenue dans cette étude consiste à redémarrer la recherche avec une nouvelle fonction de capacité ρ lors de chaque itération. En utilisant la représentation de Möbius, cela peut être réalisé à partir de la capacité ρ_1^m (cf. tableau 3.2) en diminuant la valeur de Δ de α tout en augmentant la valeur de $\Delta\ \Lambda$ de α à chaque fois qu'une nouvelle solution est identifiée et ce jusqu'à ce que la capacité obtenue soit ρ_2^m. De façon analogue, le passage de ρ_2^m à ρ_3^m est réalisé en diminuant la valeur de $\Delta\ \Lambda$ de α tout en augmentant la valeur de Λ de α à chaque fois qu'une nouvelle solution est identifiée. En procédant ainsi, chaque solution est généralement plus difficile à trouver que la précédente mais a de grandes chances d'être meilleure au regard du modèle de préférences considéré.

	Δ	$\Delta\ \Lambda$	Λ
ρ_1^m	1	0	0
ρ_2^m	0	1	0
ρ_3^m	0	0	1

TABLEAU 3.2 – Illustration de l'évolution de la transformation de Möbius de la capacité ρ. Δ et Λ sont les critères associés aux objectifs et aux préférences.

Il convient de remarquer que cette approche exploite pleinement le pouvoir expressif de l'opérateur d'agrégation qu'est l'intégrale de Choquet. En effet, la procédure proposée pour construire la représentation de Möbius de ρ considère une interaction positive $I_{\Delta,\Lambda} > 0$ entre les critères d'objectifs et de préférences modélisant ainsi une complémentarité entre ces derniers. Ainsi, toutes choses égales par ailleurs, les solutions équilibrées qui sont performantes à la fois sur les critères d'objectifs et de préférences obtiennent un meilleur score que les solutions non équilibrées. De plus,

l'approche retenue permet de construire dynamiquement la fonction de capacité ρ ce qui constitue un avantage certain. En effet, il est relativement difficile de déterminer l'équilibre entre objectifs et préférences permettant de conduire la recherche le plus efficacement possible puisque ce dernier peut varier en fonction du problème considéré. Il reste néanmoins à préciser la valeur du paramètre α. Intuitivement, plus α est grand et plus la difficulté du problème augmente rapidement lors de chaque itération. Il convient donc de choisir une valeur relativement faible tout en s'assurant que le gain observé entre deux solutions consécutives n'en devienne pas pour autant négligeable. Sur la base des expérimentations et tests réalisés, la valeur empirique $\alpha = 0.1$ a été retenue. La fonction `nextIteration()` de la stratégie de recherche SR_1 implémente l'approche qui vient d'être décrite.

Input : s_0 état initial du problème
Data : closeList liste des nœuds explorés
openList liste ordonnée des nœuds à explorer

Object SELECTIONRULE

Function `nextIteration(`s_0`)`

```
    UPDATEHEURISTICCAPACITY()
    openList ← newSortedList()
    closeList ← newList()
    openList.add(s0)
```

Algorithme 3.4 : Implémentation de `nextIteration()` pour la stratégie de recherche SR_1. `UPDATEHEURISTICCAPACITY()` met à jour la capacité ρ de l'heuristique conformément à la procédure précédemment mentionnée.

La stratégie de recherche SR_2 est une variante de SR_1 qui exploite les différences structurelles entre la fonction objectif C_μ qui sert à évaluer les solutions et les heuristiques h_1, h_2 et h_3 qui servent à guider la recherche. En effet, il convient de mentionner que l'ordre des solutions retournées par les trois heuristiques proposées ne préjugent pas nécessairement de l'ordre de ces solutions par rapport à la relation

de préférences $\succsim$ du problème. Ceci s'explique notamment par le fait que les heuristiques h_1, h_2 et h_3 utilisent des éléments extérieurs à la fonction C_μ (à savoir les considérations relatives aux objectifs à atteindre) et ne considèrent que la qualité locale des nœuds dans lesquels elles sont évaluées. En conséquence, si x_1, x_2 et x_3 sont les trois premières solutions retournées par une recherche de type BFS basée sur l'une des heuristiques proposées, le classement de ces solutions par rapport à la relation de préférences du problème pourrait être $x_3 \succsim x_1 \succsim x_2$. Il en résulte que redémarrer la recherche avec une nouvelle fonction de capacité ρ à chaque fois qu'une nouvelle solution est trouvée peut empêcher l'algorithme d'identifier des solutions potentiellement intéressantes (x_3 dans l'exemple précédent). La stratégie de recherche SR_2 est donc implémentée de sorte à ne redémarrer la recherche avec une nouvelle fonction de capacité qu'après que β solutions aient été identifiées à partir de la capacité ρ actuellement utilisée. La valeur $\beta = 10$ a été retenue sur la base d'observations empiriques. En effet, cette valeur semble être un bon compromis entre amélioration de la qualité des solutions et augmentation du temps de calcul.

Input : s_0 état initial du problème
Data : closeList liste des nœuds explorés
openList liste ordonnée des nœuds à explorer

Object SELECTIONRULE

Function `nextIteration`(s_0)
    solutionNumber ← solutionNumber + 1
    if solutionNumber mod 10 = 0 **then**
        `UPDATEHEURISTICCAPACITY()`
        openList ← `newSortedList()`
        closeList ← `newList()`
        openList.`add`(s_0)
    end

Algorithme 3.5 : Implémentation de `nextIteration()` pour la stratégie de recherche SR_2. `UPDATEHEURISTICCAPACITY()` met à jour la capacité ρ de l'heuristique conformément à la procédure précédemment mentionnée.

Les règles de sélection de l'algorithme 3.1 sont construites à partir d'une heuristique et d'une stratégie de recherche. Trois heuristiques h_1, h_2 et h_3 ainsi que deux stratégies de recherche SR_1 et SR_2 ont été proposées. Les six configurations correspondantes SR_1-h_1, SR_1-h_2, SR_1-h_3, SR_2-h_1, SR_2-h_2 et SR_2-h_3 sont évaluées dans la section 3.3.

3.2.3 Règles de coupe pour la planification avec préférences

Etant donné s un nœud du graphe (un état du système Σ) et a un arc du graphe (une action close du problème de planification), une règle de coupe a pour objectif de déterminer si l'état s' obtenu par exécution de a dans s doit être considéré dans la suite de la recherche (c.-à-d. ajouté à la frontière) ou non. Cette section présente les deux *règles d'applicabilité* de l'algorithme 3.1 ainsi que la règle de coupe dite de *Pareto-dominance*. Conceptuellement, les règles d'applicabilité peuvent être interprétées comme des règles de coupe dont la présence est obligatoire pour que les solutions retournées par l'algorithme 3.1 soient des plans valides.

Règles d'applicabilité

Par définition (cf. section 1.1.3), une action a est applicable dans un état s si $s \models Pre_a$ c'est à dire si les préconditions de a sont vérifiées dans l'état s. La première règle d'applicabilité consiste simplement à vérifier que cette condition soit respectée et à couper le nœud s' dans le cas contraire. Le nœud s' n'est donc pas ajouté à la frontière et ne peut plus être considéré au cours de la recherche.

La deuxième règle d'applicabilité s'assure que les objectifs de trajectoire ne soient pas violés au cours de la recherche. Ainsi la transition $\gamma(s, a) = s'$ n'est admissible que si pour tout objectif de trajectoire Φ du problème, $\langle s_0, \cdots, s' \rangle \models \Phi$. Si cette condition n'est pas respectée, le nœud s' n'est pas ajouté à la frontière de la recherche. L'algorithme 3.6 précise l'implémentation de ces deux règles d'applicabilité.

Input : node nœud courant de la recherche
action action à appliquer dans node
Data : preconditionTest résultat du test sur les préconditions
trajectoryTest résultat du test sur les objectifs de trajectoire
G_T ensemble des objectifs de trajectoire
Φ objectif de trajectoire

Function `IsApplicable`(node, action)

```
preconditionTest ← true
trajectoryTest ← true
if !( node.getState() ⊨ action.getPreconditions() ) then
    preconditionTest ← false
end
for all Φ ∈ G_T do
    if !( node.getTrajectory() ⊨ Φ ) then
        trajectoryTest ← false
    end
end
return preconditionTest and trajectoryTest
```

Algorithme 3.6 : Fonction `IsApplicable()` de l'algorithme 3.1

Règle de Pareto-dominance

Cette règle de coupe repose comme son nom le suggère sur le concept de *dominance au sens de Pareto* ; lequel est introduit par la définition 3.16.

Définition 3.16 - Dominance au sens de Pareto [52]
Soit P un ensemble d'attributs, $x^a, x^b \in X$ deux solutions et $y^a, y^b \in Y$ leurs vecteurs respectifs dans $Y = \{ z(x) \mid x \in X \}$.

- x^a domine fortement x^b ($x^a \succ_P x^b$) si et seulement si $\forall k \in P, x^a_k > x^b_k$;
- x^a domine faiblement x^b ($x^a \succ_p x^b$) si et seulement si $\forall k \in P, x^a_k \geq x^b_k$ et $\exists k \in P, x^a_k \neq x^b_k$.

La règle de Pareto-dominance consiste à couper le nœud s' obtenu par l'application de a dans s si ce dernier est dominé par un nœud de la frontière. Afin de s'assurer que l'évaluation de la règle de coupe puisse être réalisée efficacement, seuls les 25 premiers nœuds de la frontière sont considérés. En outre, avant de vérifier qu'un nœud de la frontière s_f domine le nœud s', il convient de s'assurer que sa structure soit similaire à celle de s'. Dans le cas contraire, il pourrait exister une action a' applicable dans s' mais pas dans s_f. Couper le nœud s' supprimerait alors une partie de l'espace de recherche pouvant contenir des solutions.

Deux états s_f et s' ont une structure similaire s'ils sont caractérisés par le même ensemble d'atomes ($Atm_f = Atm'$) et si les trajectoires $\langle s_0, \cdots, s_f \rangle$ et $\langle s_0, \cdots, s' \rangle$ sont indifférenciées au sens des objectifs et préférences de trajectoire du problème. Ce dernier point peut être vérifié à l'aide des automates de Büchi des objectifs et préférences de trajectoire PDDL (cf section 3.2.2). En effet, l'état de ces automates dans s_f et s' tient, par construction, compte de l'ensemble des actions présentes dans les trajectoires $\langle s_0, \cdots, s_f \rangle$ et $\langle s_0, \cdots, s' \rangle$. Ainsi, il suffit de s'assurer que tous les états des automates de Büchi des objectifs et préférences de trajectoire du problème aient la même valeur dans s_f et s'.

```
Input : node nœud courant de la recherche
        action action à appliquer dans node
Data :  openList liste ordonnée des nœuds à explorer
        fnode nœud de la frontière

Function IsActionPrunable(node, action)
    for i ← 1 to Max(25, openList.size()) do
        fnode ← openList.getElement(i)
        if SameStructure(fnode, node) = true then
            if StrongParetoDominance(fnode, node) = true then
                return true
            end
        end
    end
    return false
```

Algorithme 3.7 : Fonction `IsActionPrunable` de la règle de Pareto-dominance

Domaine de la gestion de crise
Malgré son apparente complexité, la démarche présentée pour résoudre les problèmes de planification avec préférences est relativement intuitive. En effet, la méthode de recherche en avant (qui consiste à identifier une progression de l'état initial vers un état final vérifiant les objectifs) est analogue aux processus cognitifs couramment employés par les décideurs de la gestion de crise. Par ailleurs, la stratégie de recherche utilisée est relativement naturelle puisqu'elle consiste, dans un premier temps, à chercher des plans satisfaisant les objectifs puis à prendre en compte les préférences des décideurs de manière progressive afin d'identifier de meilleurs solutions. Finalement, les règles de coupe sont utilisées pour éliminer les pistes de recherche qui semblent les moins prometteuses.

3.3 Implémentation et résultats expérimentaux

Un planificateur nommé CHOPLAN a été implémenté afin d'évaluer les mécanismes et algorithmes mentionnés dans la section 3.2. Cette section précise le fonctionnement de CHOPLAN et expose la démarche retenue pour le comparer aux planificateurs de l'art (cf. section 3.3.1). Par ailleurs, les résultats expérimentaux obtenus sont présentés et analysés dans la section 3.3.2.

3.3.1 Implémentation et démarche expérimentale

CHOPLAN (« CHO » pour Choquet et « PLAN » pour planificateur) a été développé à l'aide du langage de programmation Java. Il repose sur trois modules (*lexeur*, *parseur* et *solveur*) qui sont invoqués successivement lors de la résolution d'un problème de planification. Le *lexeur* traite les fichiers textes du problème (`domain.pdll` et `problem.pddl`) afin de les convertir en séquences d'entités lexicales. Ces éléments syntaxiques sont alors analysés par le *parseur* qui leur associe une sémantique et crée les objets informatiques correspondants. Ces derniers sont naturellement adaptés à la méthode de résolution choisie (par exemple des nœuds caractérisés par des atomes logiques dans le cas de la recherche guidée par des heuristiques).

Le *solveur* s'appuie sur ces objets informatiques pour implémenter l'algorithme de résolution du problème. Dans le cas de CHOPLAN, il s'agit évidemment de l'algorithme 3.1 présenté dans la section 3.2.

CHOPLAN a été implémenté à l'aide de la librairie PDDL4J [115] qui fournit un lexeur capable de traiter les versions 1.2, 2.1 et 3 du langage PDDL. De plus cette librairie propose un parseur pour la version 1.2 du langage PDDL. Ce dernier est particulièrement intéressant puisqu'il s'appuie sur les mécanismes d'inertie présentés dans [90]. La notion d'inertie permet de simplifier les différentes formules du problème ce qui se traduit généralement par une diminution de l'espace de recherche à explorer. Dans le cadre de cette étude, tous les éléments (non fournis par la librairie PDDL4J) nécessaires à l'implémentation de l'algorithme 3.1 ont été développés comme indiqué sur la figure 3.10. Ceci inclut notamment la prise en charge de l'exigence `maut-preferences` présentée dans la section 3.1.

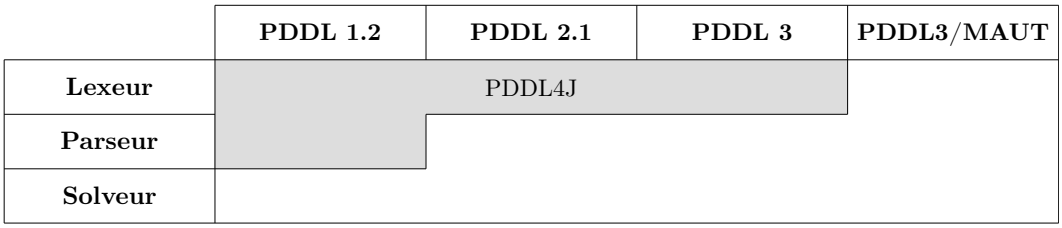


FIGURE 3.10 – Planificateur CHOPLAN. Sur fond gris, les fonctionnalités issues de la librairie PDDL4J et sur fond blanc les fonctionnalités développées.

Les performances de CHOPLAN sont évaluées en utilisant des problèmes de référence issus des compétitions internationales de planifications (IPC). Seuls les domaines de planification avec préférences qui sont susceptibles d'utiliser à la fois des préférences finales et des préférences de trajectoires PDDL3 ont été considérés. Les domaines *Rovers*, *Openstacks*, *Pathways*, *PipesWorld*, *Storage*, *TPP* et *Trucks* (voir [65] pour une description précise) sont les seuls à respecter cette condition. Dans le cadre de cette étude, ce sont les domaines *Rovers* et *Openstacks* qui ont été retenus puisque les autres domaines utilisent l'exigence PDDL `existential-preconditions` qui n'est pour le moment pas supportée dans CHOPLAN. Pour chacun de ces domaines, il existe 20 problèmes contenant uniquement des préférences finales (dénotés SP pour « Simple Preferences ») et 20 problèmes contenant des préférences finales et des préférences de trajectoire (dénotés QP pour « Qualitative Preferences »).

Comme la majorité des problèmes de référence utilisés, ces problèmes utilisent une seule préférence numérique. Afin de tester les performances de CHOPLAN sur des problèmes impliquant plusieurs préférences numériques (entre 2 et 16), une variante du domaine *Rovers* a été élaborée. Ainsi 10 problèmes (dénotés MP pour « MAUT Preferences ») contenant des préférences finales, des préférences de trajectoire ainsi que des préférences numériques ont été proposés. En conséquence, l'évaluation de CHOPLAN s'appuie sur 90 problèmes de planification avec préférences.

Les problèmes considérés sont résolus avec les six configurations SR_1-h_1, SR_1-h_2, SR_1-h_3, SR_2-h_1, SR_2-h_2 et SR_2-h_3 de CHOPLAN. Les résultats ainsi obtenus sont comparés à ceux des planificateurs de l'art SGPlan5 [83] et LPRPG-P [39]. SGPlan5 a remporté en 2006 la $5^{\text{ème}}$ compétition internationale de planification dans la catégorie planification avec préférences. Il s'agit de la seule compétition au cours de laquelle la résolution des problèmes de planification avec préférences *Rovers* et *Openstacks* a été proposée. La méthode de résolution de SGPlan5 consiste à subdiviser le problème en sous-problèmes qui sont résolus indépendamment les uns des autres. Ces mécanismes de division sont optimisés pour chacun des domaines considérés. Ainsi, une variante nommée SGPlan-W est également retenue afin d'évaluer les capacités de résolution de SGPlan5 en tant que solveur générique. Cette dernière consiste simplement à invoquer SGPlan5 en lui fournissant des problèmes dont les noms initiaux ont été modifiés (voir [39] pour plus de détails). LPRPG-P est quant à lui le planificateur de l'art le plus récent. Il est capable de traiter l'ensemble des préférences PDDL3 et peut gérer des préférences numériques. Sa méthode de résolution couple des mécanismes de programmation linéaire avec une recherche heuristique basée sur un graphe de planification relaxé. Par ailleurs, l'ensemble des problèmes considérés a également été résolu avec une version de ChoPlan basée uniquement sur l'estimation Δ_1. Ce planificateur nommé *Contrôle* constitue une implémentation de l'heuristique FAST-FORWARD dans CHOPLAN. Ne cherchant pas à optimiser la fonction objectif, il fournit des valeurs de référence pour estimer le gain qualitatif apporté par les planificateurs CHOPLAN, SGPlan5, SGPlan-W et LPRPG-P.

Tous les tests ont été réalisés sur la même machine (3.2Ghz CPU, 8Go RAM). Pour chacun des 90 problèmes considérés, les planificateurs ont disposé de 10 minutes de temps de calcul. La qualité des plans solutions a été calculée à l'aide du validateur

de plan VAL [82]. Cet outil est utilisé lors des compétitions internationales de planification afin de vérifier la validité des solutions et d'évaluer leurs scores par rapport à la fonction objectif du problème considéré. Son utilisation permet de comparer CHOPLAN aux planificateurs de l'art de manière cohérente et ce même si celui-ci résout une version PDDL3/MAUT des problèmes PDDL considérés.

3.3.2 Résultats expérimentaux

Les figures 3.11 à 3.15 représentent la qualité des solutions obtenus par Contrôle, LPRPG-P, SGPlan5, SGPlan-W et CHOPLAN pour l'ensemble des problèmes étudiés. Dans tous les problèmes considérés, la fonction objectif doit être *minimisée*. Ainsi, plus la valeur associée à la solution d'un planificateur est faible et plus ce dernier est performant. Dans le cas où un planificateur retourne plusieurs solutions, seule la meilleure est représentée. Une absence de valeur pour un problème donné signifie que le planificateur n'a pas réussi à résoudre le problème concerné. Dans le but de ne pas surcharger la présentation de ces résultats, une seule des six configurations de CHOPLAN a été représentée sur chaque figure. La configuration retenue est celle ayant obtenue les meilleurs résultats sur le domaine considéré. Les performances des différentes configurations de CHOPLAN peuvent néanmoins être analysées à l'aide des tableaux 3.3, 3.4 et 3.5 qui synthétisent l'ensemble des résultats obtenus.

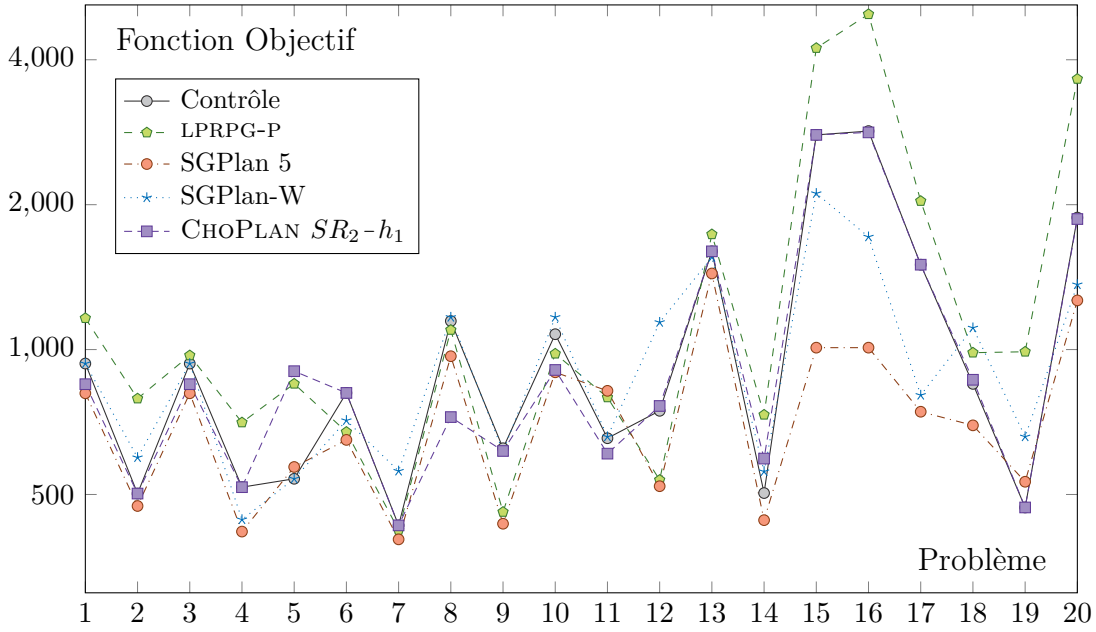


FIGURE 3.11 – Résultats pour *Rovers - Simple Preferences*

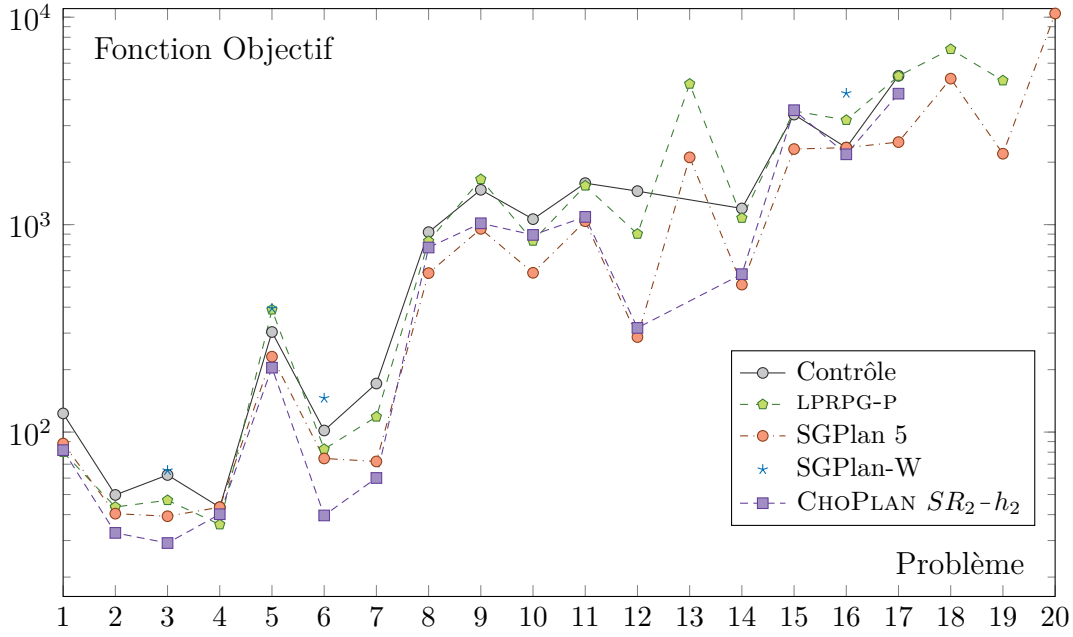


FIGURE 3.12 – Résultats pour *Rovers - Qualitative Preferences*

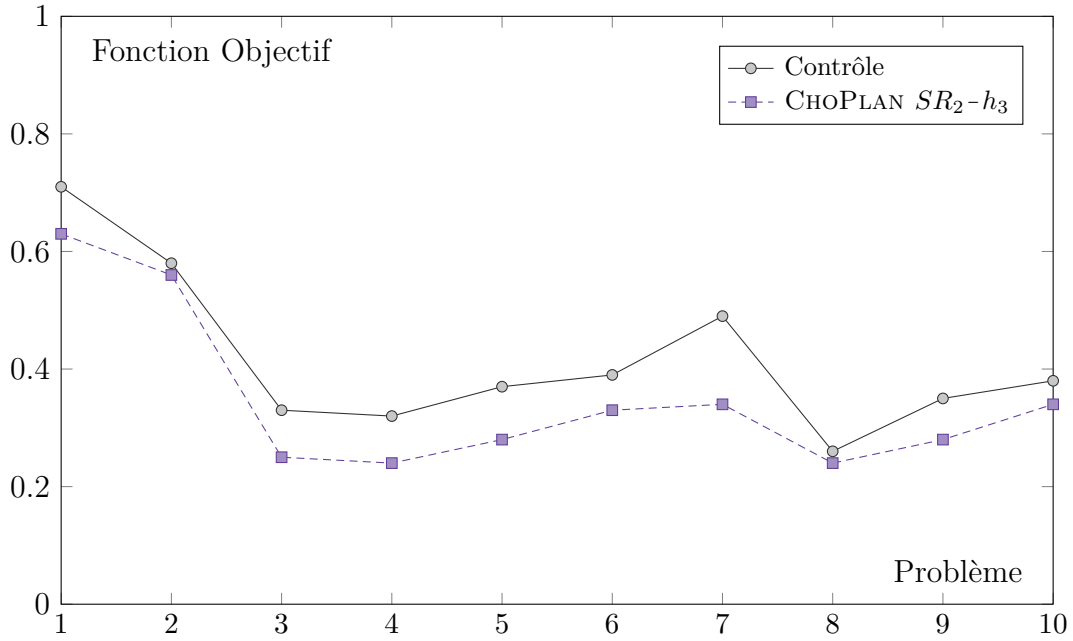


FIGURE 3.13 – Résultats pour *Rovers* - MAUT *Preferences*

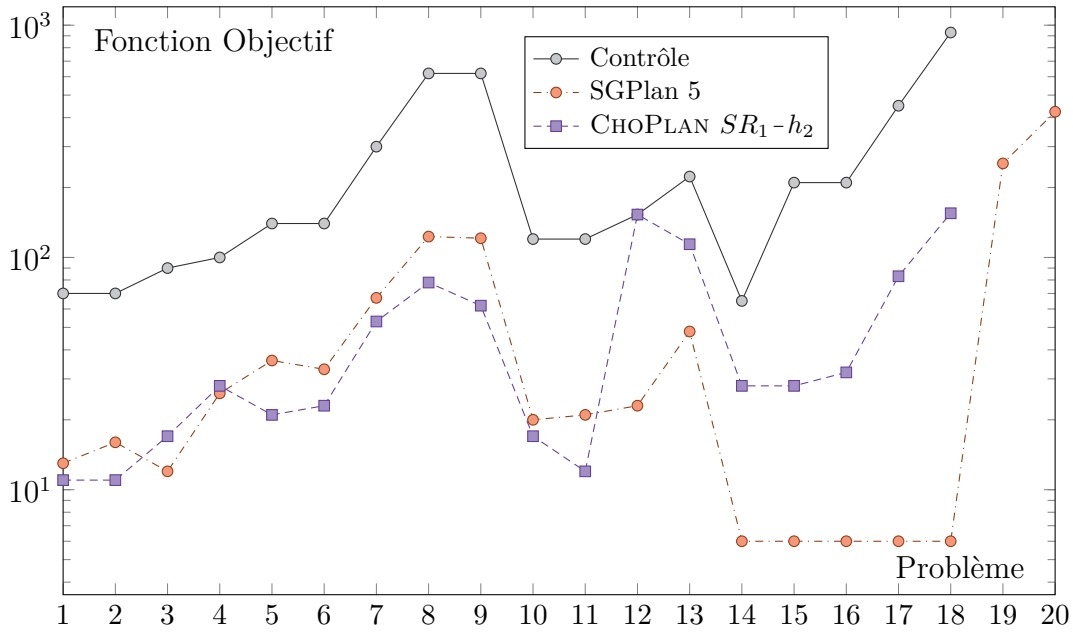


FIGURE 3.14 – Résultats pour *Openstacks - Simple Preferences*

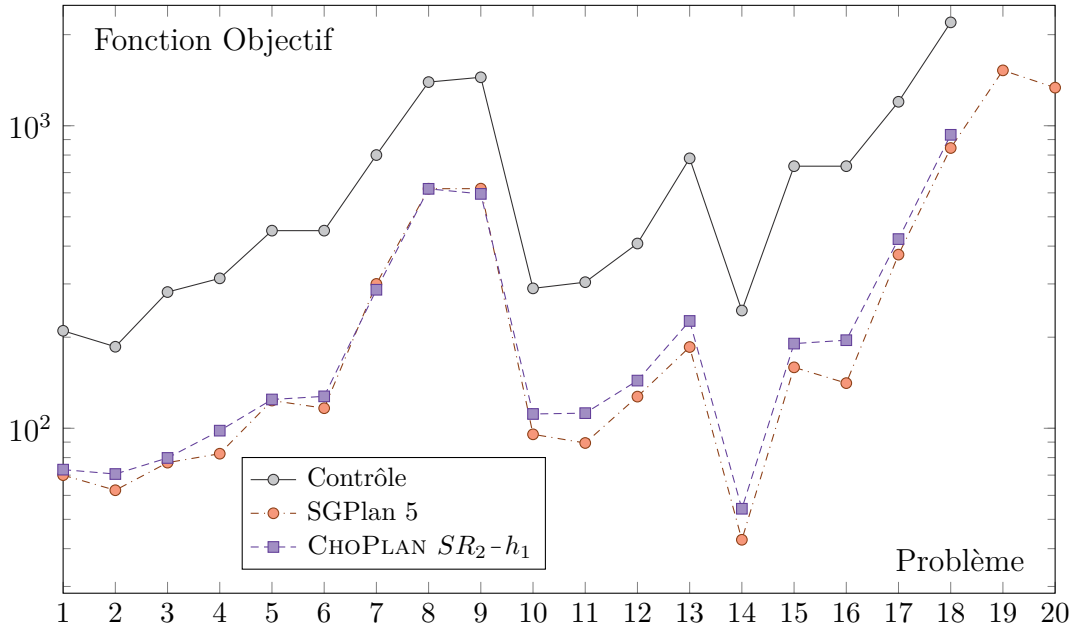


FIGURE 3.15 – Résultats pour *Openstacks - Qualitative Preferences*

Le tableau 3.3 présente le nombre de problèmes résolus par chacun des planificateurs considérés ainsi que l'efficacité de ces derniers. L'efficacité est définie comme le nombre de fois qu'un planificateur résout un problème en trouvant une solution dont la qualité est supérieure ou égale à celle de la solution retournée par le planificateur de Contrôle. Sans surprise, les problèmes de type *Rovers - MP* qui sont exprimés selon le formalisme PDDL3/MAUT n'ont été résolus que par les planificateurs dérivés de CHOPLAN.

Le planificateur SGPlan5 exhibe des performances remarquables puisqu'il est capable de résoudre tous les problèmes SP et QP (80/90) tout en obtenant un score d'efficacité élevé (77/90). Ces performances semblent néanmoins fortement liées aux optimisations réalisées spécifiquement pour le domaine considéré puisque SGPlan-W obtient quant à lui les moins bons résultats avec seulement 24 problèmes résolus.

Le nombre de solutions identifiées par le planificateur LPRPG-P est moyen (39/90) ce qui s'explique par l'incapacité de ce dernier à traiter les problèmes *Openstacks* (à cause d'une exigence PDDL non supportée). Toutefois, il convient de préciser que les tests réalisés ne reflètent pas pleinement le potentiel de LPRPG-P. En effet, ce dernier peut par ailleurs résoudre les problèmes *Pathways*, *Storage*, *TPP* et *Trucks* qui n'ont pas été retenus dans le cadre de cette étude.

CHOPLAN obtient quant à lui de bons résultats puisqu'il est capable de résoudre 82 des 90 problèmes considérés et ce avec une bonne efficacité (entre 82 et 86 selon la configuration employée). Les heuristiques utilisées par CHOPLAN peuvent être considérées comme des variantes de FAST-FORWARD qui consistent à faire dériver la recherche pour prendre en compte progressivement les préférences du problème. En conséquence, il est normal que CHOPLAN trouve exactement le même nombre de solutions que le planificateur Contrôle.

Ces premiers résultats rendent compte des performances propres des planificateurs (nombre de solutions trouvées et nombre de fois qu'un planificateur identifie une meilleure solution que celle retournée par Contrôle) sans pour autant comparer directement les planificateurs les uns par rapport aux autres.

Nombre Solutions & Efficacité	Contrôle	LPRPG-P	SGPlan 5	SGPlan-W	ChoPlan SR_1-h_1	ChoPlan SR_1-h_2	ChoPlan SR_1-h_3	ChoPlan SR_2-h_1	ChoPlan SR_2-h_2	ChoPlan SR_2-h_3
Rov. SP	20	20	20	20	20	20	20	20	20	20
Rov. QP	16	19	20	4	16	16	16	16	16	16
Rov. MP	10	-	-	-	10	10	10	10	10	10
OpS. SP	18	-	20	0	18	18	18	18	18	18
OpS. QP	18	-	20	0	18	18	18	18	18	18
Solutions	**82/90**	**39/90**	**80/90**	**24/90**	**82/90**	**82/90**	**82/90**	**82/90**	**82/90**	**82/90**
Rov. SP	-	6	17	11	15	14	13	16	14	13
Rov. QP	-	16	20	4	20	19	19	20	19	19
Rov. MP	-	0	0	0	10	10	10	10	10	10
OpS. SP	-	2	20	2	20	20	20	20	20	20
OpS. QP	-	2	20	2	20	20	20	20	20	20
Efficacité	**-**	**26/90**	**77/90**	**19/90**	**85/90**	**83/90**	**82/90**	**86/90**	**83/90**	**82/90**

TABLEAU 3.3 – Nombre de solutions identifiées et éfficacité des planificateurs

La notion de score IPC (qui comme son nom le suggère est utilisée lors des compétitions internationales de planification) permet d'analyser comparativement les performances d'un ensemble de planificateurs Π. Le score d'un planificateur π pour un problème p est défini par :

$$\mathit{score}\,(\pi, p) \;=\; \mathit{meilleure\text{-}qualité}\,(p) \;\div\; \mathit{qualité}\,(\pi, p)$$

La fonction $\mathit{meilleure\text{-}qualité}\,(p)$ retourne la qualité de la meilleure solution trouvée par l'ensemble des planificateurs de Π tandis que la fonction $\mathit{qualité}\,(\pi, p)$ désigne la qualité de la solution identifiée par π. Si le planificateur π n'a pas résolu le problème p, alors $\mathit{score}\,(\pi, p) = 0$. Le score IPC de π pour un ensemble P de problèmes vaut :

$$\mathit{score\text{-}IPC}\,(\pi, P) \;=\; \sum_{p \in P} \mathit{score}\,(\pi, p)$$

Le score IPC est un indicateur intéressant puisqu'il tient compte de la différence de qualité entre les solutions des planificateurs (en les comparant à la meilleure trouvée) tout en récompensant les planificateurs capables de résoudre un grand nombre de problèmes. Le tableau 3.4 présente les scores IPC des planificateurs étudiés. Les tendances observées avec le tableau 3.3 sont confirmées puisque SGPlan 5 obtient le meilleur score et SGPlan-W le moins bon tandis que les résultats de LPRPG-P sont moyens.

Avec des scores IPC compris entre 57,75 et 61,41 (soit un gain d'environ 50 % par rapport au planificateur de Contrôle), CHOPLAN exhibe des performances tout à fait satisfaisantes. Par ailleurs, les six configurations considérées ont des scores relativement similaires. Ceci semble indiquer que le mécanisme retenu (agrégation d'estimations à l'aide d'une intégrale de Choquet avec modification de la capacité lors de chaque itération) joue un rôle prépondérant dans les performances globales de CHOPLAN. Néanmoins, des différences notables entre configurations peuvent être observées pour certains domaines. Par exemple, SR_2 - h_2 est meilleure que SR_2 - h_1 sur *Openstacks - SP* mais moins efficace sur *Openstacks - QP* (voir tableau 3.5 pour une analyse détaillée).

Score IPC	Contrôle	LPRPG-P	SGPlan 5	SGPlan-W	CHOPLAN SR_1 -h_1	CHOPLAN SR_1 -h_2	CHOPLAN SR_1 -h_3	CHOPLAN SR_2 -h_1	CHOPLAN SR_2 -h_2	CHOPLAN SR_2 -h_3
Rov. SP	15,49	13,30	19,32	15,56	15,22	15,45	14,95	15,71	15,50	14,94
OpS. SP	2,09	0	17,00	0	4,24	11,21	7,54	4,31	10,93	7,66
Tot. SP	**17,58**	**13,30**	**36,31**	**15,56**	**19,46**	**26,66**	**22,49**	**20,02**	**26,43**	**22,60**
Rov. QP	8,94	11,53	18,06	1,54	13,34	13,52	13,31	13,49	13,74	13,46
OpS. QP	5,39	0	19,86	0	15,88	11,66	12,17	15,99	10,92	12,14
Tot. QP	**14,32**	**11,53**	**37,92**	**1,54**	**29,22**	**25,18**	**25,48**	**29,49**	**24,66**	**25,61**
Rov. MP	8,10	0	0	0	9,25	9,57	9,79	9,28	9,57	9,82
Total	**40,01**	**24,83**	**74,23**	**17,09**	**57,93**	**61,41**	**57,75**	**58,79**	**60,66**	**58,03**

TABLEAU 3.4 – Score IPC des planificateurs

Le tableau 3.5 synthétise les résultats obtenus par CHOPLAN de sorte à étudier l'impact des stratégies de recherche SR_1 et SR_2 ainsi que celui des heuristiques h_1, h_2 et h_3. Pour cela, tous les résultats impliquant SR_1 et SR_2 (respectivement h_1, h_2 et h_3) ont été additionnés et ce quelle que soit l'heuristique employée (respectivement la stratégie de recherche employée) puis triés en fonction du type de problème considéré (SP, QP ou MP).

Au vu de la très faible différence entre les scores de SR_1 - h_* et SR_2 - h_*, il apparait clairement que les performances de CHOPLAN sont identiques quelque soit la stratégie de recherche employée. Pour le temps de calcul considéré (10 minutes par problème), la variante SR_2 n'apporte donc pas de gain significatif quant à la qualité des solutions identifiées. Toutefois, il convient de préciser que le nombre de solutions intermédiaires identifiées par la stratégie SR_2 (non visible dans les résultats présentés qui ne retiennent que la meilleure solution) est généralement supérieur à celui de la stratégie SR_1. En conséquence, bien que l'apport de SR_2 par rapport à SR_1 soit négligeable dans le cas général, ce dernier pourrait être plus significatif pour des temps de calcul faible (de l'ordre de la seconde).

En ce qui concerne les heuristiques h_1, h_2 et h_3, les résultats sont un peu plus contrastés. L'heuristique h_2 qui associe une interprétation floue aux préférences PDDL3 est celle qui obtient les meilleures performances. Elle surpasse grandement h_1 sur les problèmes ne contenant que des préférences finales ce qui s'explique intuitivement par le fait que h_2 est une heuristique plus informée que h_1. De plus, elle obtient également de meilleurs résultats sur les problèmes de type MP bien que la différence observée soit plus faible. En revanche, l'heuristique h_1 est plus performante dans le cas des problèmes QP qui, pour rappel, contiennent des préférences finales et des préférences de trajectoires. Ce résultat peut sembler surprenant et son interprétation n'est a priori pas aisée. Il s'explique peut être par le traitement différent que subissent, dans l'heuristique h_2, les préférences `always` et `at-most-once` d'une part et les préférences `sometime` et `sometime-before` d'autre part. En effet, ce dernier pousse l'heuristique h_2 à être plus optimiste ou pessimiste au regard des préférences `always` et `at-most-once` (qui sont jugées soient vraies soient fausses) qu'elle ne l'est pour les préférences `sometime` et `sometime-before`

(qui sont jugées en fonction de l'effort à fournir pour les satisfaire). Ceci entraine implicitement une modification du calcul de l'utilité dont le résultat n'est alors plus entièrement représentatif des préférences du problème diminuant ainsi l'efficacité de l'heuristique. Une solution pourrait être de remplacer Λ_2 par deux estimations Λ_2^1 (responsable des préférences `always` et `at-most-once`) et Λ_2^2 (responsable des préférences `sometime`, `sometime-before` ainsi que des préférences finales). En agrégeant ces deux grandeurs correctement, il serait alors possible de créer une estimation Λ_2' qui ne souffrirait pas du défaut précédemment mentionné.

Le cas de l'heuristique h_3 est également intéressant. Bien qu'elle obtiennent globalement les plus mauvais résultats, il s'agit de l'heuristique la plus efficace pour les problèmes de type MP. Néanmoins, cette catégorie ne contenant que 10 problèmes, il convient de rester prudent quant à une éventuelle généralisation de ce résultat. Pour rappel, l'estimation Λ_3 (utilisée par h_3) a été construite via une somme pondérée de Λ_1 (utilisée par h_1) et de Λ_2 (utilisée par h_2). Ce comportement transparait dans le résultat final puisque les valeurs obtenues par h_3 sont comprises entre celles de h_1 et de h_2. Ainsi, il pourrait être intéressant de construire une estimation Λ_4 à partir d'une intégrale de Choquet qui représenterait une substituabilité parfaite entre Λ_1 et Λ_2. Une heuristique basée sur Δ_1 et Λ_4 combinerait peut être les avantages de h_1 et h_2 et pourrait ainsi être très performante sur l'ensemble des problèmes considérés.

Score IPC ChoPlan	SR_1 - h_*	SR_2 - h_*	SR_* - h_1	SR_* - h_2	SR_* - h_3
Tot. SP	68,60	69,06	39,49	53,09	45,09
Tot. QP	79,88	79,75	58,71	49,84	51,08
Tot. MP	28,61	28,67	18,53	19,15	19,60
Total	**177,09**	**177,48**	**116,72**	**122,08**	**115,78**

TABLEAU 3.5 – Synthèse des scores IPC des configurations de CHOPLAN

Les figures 3.16 à 3.18 comparent le planificateur CHOPLAN et les planificateurs Contrôle, LPRPG-P, SGPlan 5 et SGPlan-W. Ces graphiques ont pour abscisse la qualité des solutions identifiées par CHOPLAN et pour ordonnée la qualité des solutions identifiées par le planificateur auquel CHOPLAN est comparé. Seuls les problèmes ayant été résolus par les deux planificateurs sont représentés. De plus, la fonction identité est également représentée afin de faciliter l'analyse. Lorsqu'un point est au dessus de la courbe de la fonction identité, cela signifie que la solution retournée par CHOPLAN pour le problème en question est meilleure que celle du planificateur auquel il est comparé. La différence de qualité entre les solutions identifiées par les planificateurs est d'autant plus grande que le point est éloigné de la courbe de la fonction identité. Afin d'apprécier correctement cette différence de qualité, il convient de remarquer que l'échelle utilisée sur ces figures est logarithmique. En outre, la configuration de CHOPLAN retenue pour cette analyse est celle ayant conduit à l'identification des meilleurs résultats à savoir SR_1 - h_2.

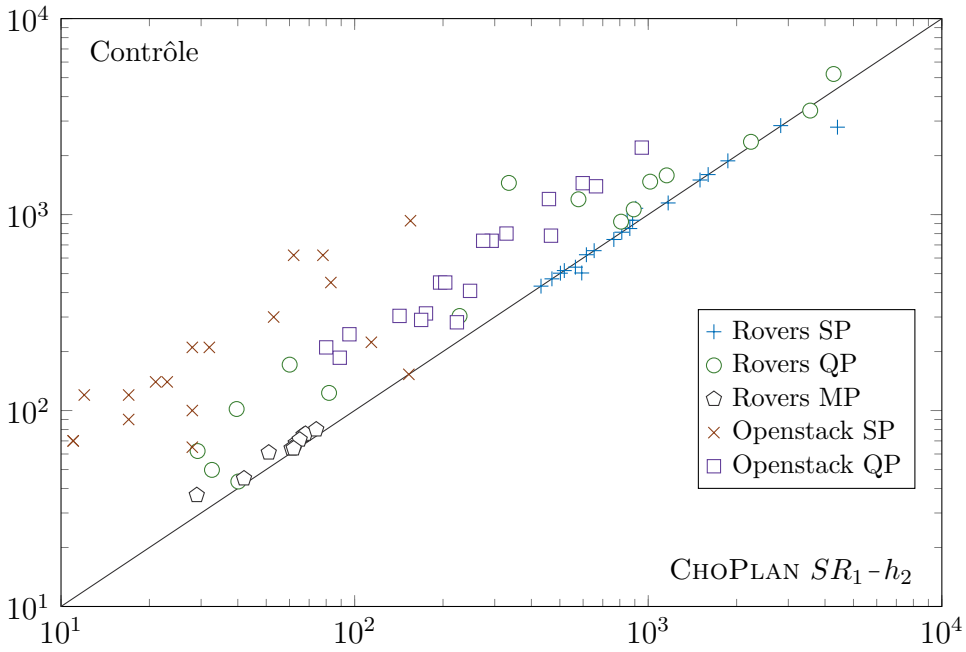


FIGURE 3.16 – Comparaison de CHOPLAN SR_1 - h_2 et de Contrôle

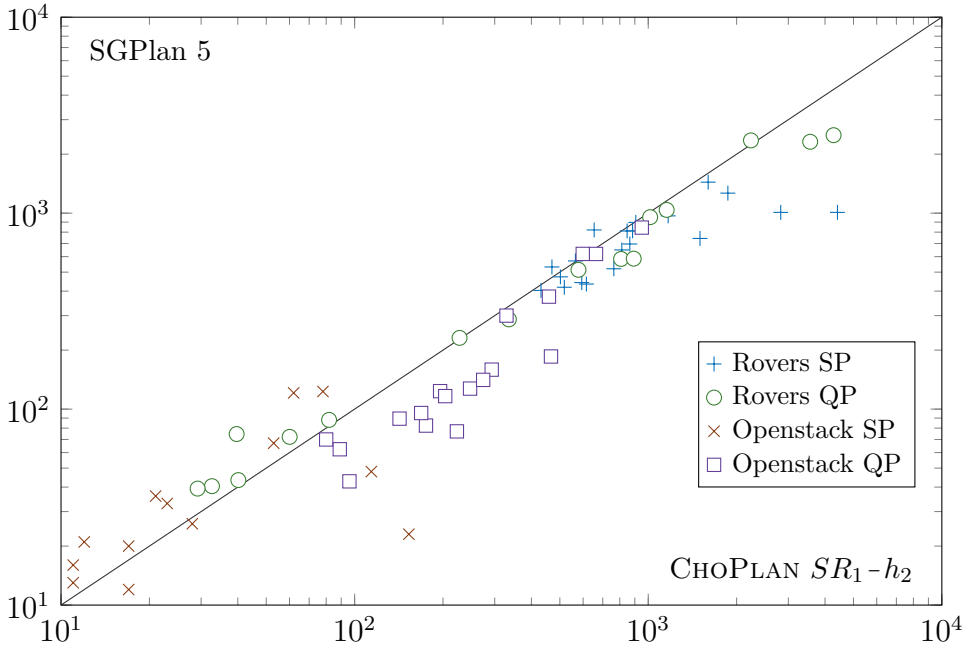


FIGURE 3.17 – Comparaison de CHOPLAN SR_1 - h_2 et de SGPlan 5

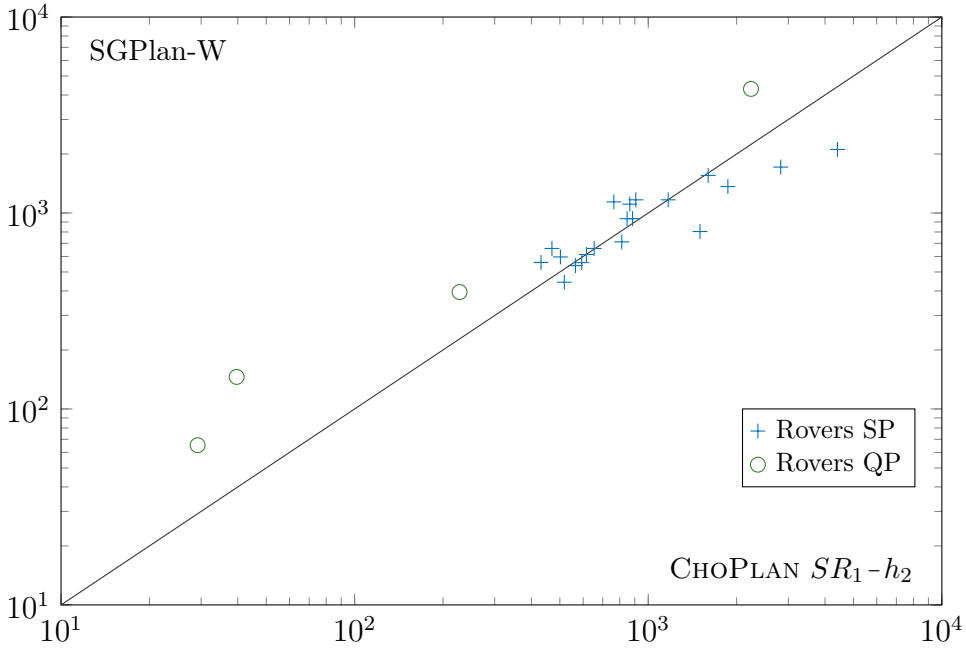


FIGURE 3.18 – Comparaison de CHOPLAN SR_1 - h_2 et de SGPlan-W

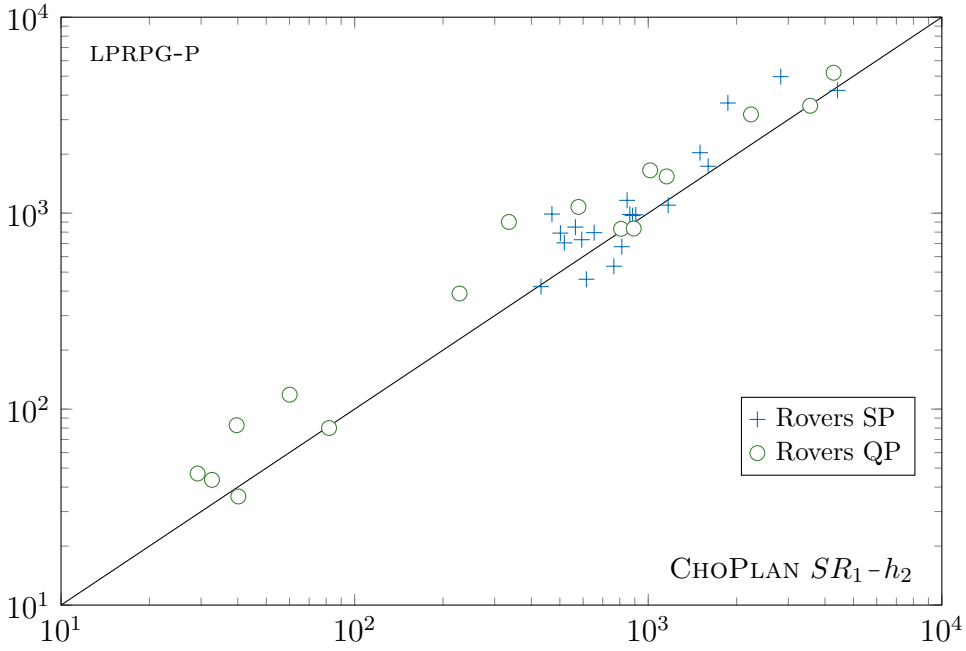


FIGURE 3.19 – Comparaison de CHOPLAN SR_1 - h_2 et de LPRPG-P

Pour finir, il reste à étudier l'impact de la règle de coupe mentionnée dans la section 3.2.3. Le tableau 3.6 présente la différence de temps de calcul observée lorsque CHOPLAN SR_1 - h_2 est utilisé avec et sans la règle de Pareto-dominance forte. Le gain observé gain varie fortement en fonction des problèmes considérés ce qui s'explique intuitivement par le fait que certains domaines sont naturellement plus prompts à engendrer des nœuds Pareto-dominés que d'autres. C'est par exemple le cas du domaine *Rovers* dans lequel un robot peut effectuer des allers-retours entre deux positions. Bien qu'une telle action puisse être inutile du point de vue de l'atteinte des objectifs, elle peut néanmoins dégrader un critère de coût de déplacement.

Pareto-F	**Gain Temps Calcul**
Rovers SP	42,65 %
Rovers QP	34,23 %
Openstack SP	10,73 %
Openstack QP	12,84 %

TABLEAU 3.6 – Impact de la règle de Pareto-dominance forte

Conclusion

Ce chapitre a débuté par la description du formalisme PDDL3/MAUT qui généralise le langage PDDL3 par l'introduction de préférences floues. Ce dernier enrichit le pouvoir expressif du PDDL3 puisqu'il permet notamment de représenter facilement plusieurs préférences numériques, de construire des préférences par agrégation d'autres préférences ou encore de tenir compte des interactions entre les différentes préférences du problème. Un algorithme itératif de résolution de problèmes de planification avec préférences a ensuite été présenté. Celui-ci s'appuie sur des heuristiques qui évaluent le potentiel d'un nœud à l'aide de deux estimations respectivement relatives aux objectifs à atteindre et aux préférences du problème. L'originalité de ces heuristiques consiste à agréger ces deux estimations à l'aide d'une intégrale de Choquet. De plus, une règle de coupe permettant d'identifier et de supprimer les nœuds qu'il n'est pas nécessaire de visiter lors de la recherche a été proposée. Le planificateur CHOPLAN repose sur ces mécanismes et a été implémenté dans le cadre de cette étude. Ses performances ont été comparées à celles de plusieurs planificateurs de l'art. Les résultats obtenus suggèrent que l'approche proposée est efficace puisque CHOPLAN identifie de meilleures solutions que celles proposées par le planificateur de Contrôle, SGPlan-W et LPRPG-P. Sur l'ensemble des problèmes considérés, seuls les résultats du planificateur SGPlan 5 (qui pour rappel utilise des mécanismes d'optimisation différents pour chaque domaine) surpassent ceux de CHOPLAN.

Chapitre 4

De l'interopérabilité des organisations en gestion de crise

Ce chapitre s'intéresse à l'utilisation des résultats exposés dans le chapitre 3 à un domaine applicatif particulier à savoir celui de la gestion de crise. En effet, les mécanismes de planification avec préférences qui y sont décrits sont ici utilisés pour contribuer à la résolution des problèmes liés à l'interopérabilité d'un ensemble d'organisations qui collaborent dans le but de résoudre une situation de crise.

La section 4.1 introduit le domaine de la gestion de crise en présentant les enjeux et défis qui y sont associés. Elle s'intéresse en particulier aux problèmes de collaboration que peuvent rencontrer les différents partenaires mobilisés lors de la résolution d'une crise. Ces derniers limitent l'efficacité de la réponse à la crise mais peuvent être adressés par la réalisation de plans d'action collaboratifs. La section 4.2 présente une démarche en trois étapes (modélisation, planification et aide à la décision) spécialement conçue pour construire de tels plans. Dans le cadre de ces travaux, un prototype logiciel basé sur cette démarche de construction a été développé. Son utilisation est illustrée dans la section 4.3 à l'aide d'un cas d'usage extrait du projet européen SECTOR [38] qui consiste à résoudre une situation de grande inondation en Europe du Nord.

4.1 De la problématique de la gestion de crise

4.1.1 Enjeux et défis associés à la gestion de crise

Une *crise* (ou *situation critique*) peut être définie comme « une situation ayant des impacts à long terme provoquée par un évènement qui a engendré des pertes et dommages importants et entrainé l'interruption d'une ou plusieurs activités critiques » [34]. De telles situations peuvent par exemple être causées par des catastrophes naturelles (tsunamis, séismes, inondations...) ou par des accidents industriels (explosion sur un site industriel...). La gestion des situations de crise représente toujours un défi majeur pour notre société. En effet, entre 2004 et 2013, les crises ont été responsables de la mort d'environ 100 000 personnes par an d'après la dernière étude statistique de l'Emergency Event Database [77].

Les situations critiques ont été largement étudiées afin d'améliorer leur gestion et de réduire leurs conséquences catastrophiques. Les crises sont souvent analysées (voir par exemple [2]) selon quatre phases de référence : *prévention*, *préparation*, *réponse* et *rétablissement*. La prévention et la préparation interviennent avant la survenue de la crise et ont respectivement pour but d'empêcher l'apparition de la crise et de se préparer à la gérer si elle venait à se produire malgré les efforts précédemment réalisés. Dans de tels cas, la phase de réponse a alors pour objectif de transformer la situation de crise en une situation acceptable temporairement. Le rétablissement s'intéresse quant à lui aux actions à mener pour transformer cette situation acceptable temporairement en une situation dite de retour à la normale. Cette étude se focalise particulièrement sur la phase de *réponse*.

Lorsqu'une crise survient, de nombreuses *organisations* sont mobilisées pour y apporter une réponse. Le terme d'organisation (ou d'*entité organisationnelle*) est ici considéré dans son acception courante la plus générique. Il peut par exemple s'agir des services publics de l'Etat tels que les pompiers ou la gendarmerie mais également d'entreprises ou d'associations. Le management de cette réponse est généralement placé sous la responsabilité d'une *cellule de crise* composée des différentes parties prenantes de la gestion de crise : un ou plusieurs décideurs assistés par les acteurs et experts pertinents au vu de la situation. La mission des membres de la cellule est très délicate puisque ces derniers doivent coordonner le travail

collectif des différentes organisations tout en étant soumis à de fortes pressions temporelles et psychologiques. Par ailleurs, dans la plupart des cas, les organisations mobilisées sont relativement hétérogènes (aux niveaux culturel, fonctionnel et technologique) et peu, voire pas du tout, entrainées à travailler ensemble. Ceci génère inévitablement des problèmes de collaboration (définition collective des objectifs difficile, partage d'informations incomplet, mauvaise coordination des acteurs...) qui limitent l'efficacité des actions entreprises par la cellule de crise. Ces problèmes ont été identifiés à travers de nombreux retours d'expérience issus de crises passées [76, 135, 142] montrant ainsi que le niveau de maturité de la collaboration entre les entités organisationnelles mobilisées est l'un des facteurs limitants de l'efficacité de la gestion de crise.

La notion d'*interopérabilité* qui peut être définie comme « la capacité que possède [une organisation] à fonctionner avec d'autres [organisations] existantes ou futures » [1] permet de préciser le niveau de maturité de la collaboration d'un aréopage d'entités organisationnelles. Il est possible de distinguer plusieurs modes d'interopérabilité [134] qui caractérisent l'intensité d'une collaboration :

1. *Communication* : les organisations fonctionnant dans ce mode sont en mesure de s'échanger et de partager des informations ;
2. *Coordination* : dans ce mode d'interopérabilité, les entités organisationnelles peuvent mettre à disposition de leurs partenaires certaines de leurs compétences via la réalisation de tâches spécifiques ;
3. *Coopération* : des organisations travaillent dans ce mode d'interopérabilité si elles cherchent à atteindre un objectif commun. Ce niveau implique l'existence d'une stratégie de collaboration partagée par l'ensemble des partenaires mobilisés.

En outre, le concept d'interopérabilité possède une *dimension opérationnelle* (relative aux différents métiers des parties prenantes) et une *dimension technologique* qui peuvent être analysées séparément. L'aspect opérationnel englobe l'identification des besoins de partage d'informations et de tâches (en mode communication et coordination) ainsi que la définition de la stratégie de collaboration commune (en mode coopération). L'aspect technologique s'assure qu'une organisation est effectivement en mesure de partager certaines informations et tâches (en mode communication et

coordination) ou de s'insérer au sein d'une stratégie de collaboration commune (en mode coopération). Par la suite, lorsqu'il sera fait mention d'interopérabilité, c'est le mode coopération (qui englobe les niveaux « Communication » et « Coordination ») qui sera considéré.

Cette présentation du domaine de la gestion de crise motive la formulation de la problématique suivante : *Comment supporter et améliorer l'interopérabilité au sein d'un aréopage d'organisations mobilisées pour résoudre une situation de crise ?*

4.1.2 Plans d'action collaboratifs de gestion de crise

Pour adresser ces problèmes d'interopérabilité, des plans de gestion de crise sont traditionnellement préparés avant que la crise ne survienne (*planification à froid*). Malheureusement, ces plans sont souvent imparfaits puisque d'une part ils sont générés avant que le contexte opérationnel de la collaboration ne soit connu avec précision et que d'autre part les situations de crise réelles divergent souvent rapidement de celles qui ont été planifiées. Dwight D. Eisenhower dira à ce propos que « Les plans ne sont rien, tout est dans la planification » [53]. Au travers de cette affirmation qui peut sembler obscure de prime abord, le stratège militaire souligne l'absolue nécessité pour un plan de s'inscrire pleinement dans le contexte opérationnel qui justifie sa création. L'approche qui consiste à construire les plans de gestion de crise uniquement après la survenue de la crise (*planification à chaud*) est particulièrement intéressante puisqu'elle permet de coordonner les actions des différents partenaires mobilisés pour résoudre la situation sans souffrir des défauts de la planification à froid.

En raison du niveau de granularité auquel travaille la cellule de crise, les plans de gestion de crise sont des plans singuliers à certains égards. Ils doivent faire intervenir un grand nombre d'acteurs et se concentrent moins sur la description détaillée des actions à mener (ce qui relève du domaine de compétence des différentes organisations mobilisées) que sur l'orchestration des ces différentes actions. En effet, conformément au *principe de subsidiarité*, les décideurs de la cellule de crise précisent aux partenaires ce qu'ils doivent faire et quand ils doivent le faire sans jamais leur dire comment cela doit être réalisé. Ces précisions motivent la définition de la notion de *plan d'action collaboratif*.

Définition 4.1 - Plan d'action collaboratif [13]
Un *plan d'action collaboratif* est un processus qui orchestre les actions qui doivent être mises en œuvre par un aréopage d'entités organisationnelles pour que ces dernières puissent atteindre collectivement leurs objectifs communs.

Un plan d'action collaboratif contient des actions exécutables par les partenaires de la collaboration qui sont invoquées successivement et/ou en parallèle les unes des autres jusqu'à ce que la mission visée soit accomplie. Il doit respecter les objectifs des membres de la cellule de crise et doit, dans la mesure du possible, optimiser leurs préférences. Les plans collaboratifs peuvent être représentés par un sous-ensemble de la notation BPMN (Business Process Model and Notation) [112] comme illustré sur la figure 4.1. Ces derniers sont alors constitués d'un *pool* contenant une unique *lane* (grands rectangles gris) pour chaque partenaire de la collaboration ainsi que d'un *pool* et d'une *lane* dédiés à la cellule de crise. Les actions à exécuter (représentées par les petits rectangles vert, violet et rouge) sont contenues dans les *lanes* de leur acteur respectif. A chaque action d'un partenaire correspond une action d'invocation dans la *lane* de la cellule de crise (représentées en bleu). Celle-ci contient également des connecteurs qui permettent de synchroniser l'invocation des actions des différentes entités organisationnelles considérées.

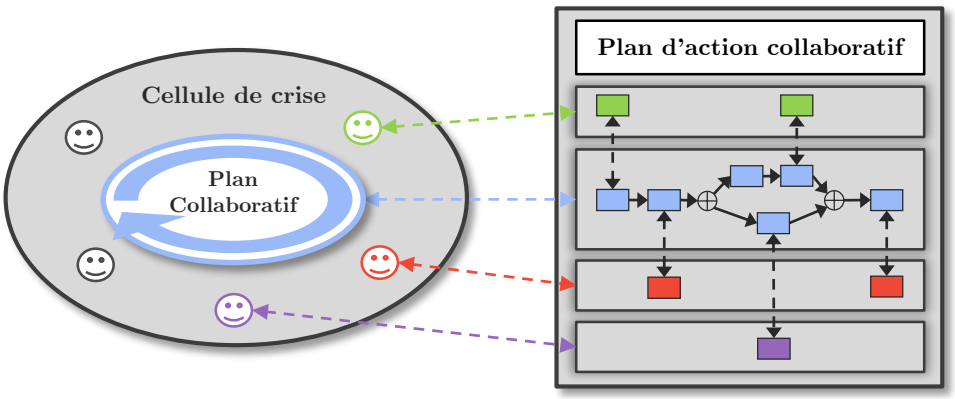


FIGURE 4.1 – Cellule de crise et plan d'action collaboratif

La construction de plans d'action collaboratifs est une problématique qui intéresse la société Thales Communications and Security dans le cadre de ses activités de construction de systèmes C4I (Command, Control, Communications, Computers and Intelligence) pour les acteurs de la sécurité. Cette dernière intéresse également le laboratoire de Génie Industriel de l'Ecole des Mines d'Albi dont le projet de recherche MISE (Mediation Information System Engineering) [15, 17, 18] adresse les problèmes d'interopérabilité des organisations. Ces deux entités ont été amenées à collaborer de 2006 à 2010 dans le cadre du projet ANR ISyCri (Interopérabilité des Systèmes en situation de Crise) [137, 139] et de 2013 à 2016 par l'intermédiaire de ces travaux de thèse.

L'approche mise en œuvre dans le projet ISyCri pour construire des plans collaboratifs de gestion de crise repose sur des mécanismes de modélisation et de déduction logique. En effet, il est demandé aux décideurs de représenter leur problème afin de constituer une base de connaissances qui est ensuite exploitée par inférence dans le but de construire un plan d'action susceptible de résoudre la situation critique considérée. Les travaux présentés dans ce chapitre s'inspirent des résultats du projet ISyCri et en étendent la portée et les applications en tenant compte des ressources à disposition et des préférences des décideurs. Pour ce faire, ils s'appuient sur le planificateur ChoPlan qui a été présenté dans le chapitre 3 (cf. section 4.2).

Pour finir, il convient de préciser que le périmètre du projet MISE dans lequel s'intègre cette étude ne se limite pas seulement à la construction du plan d'action collaboratif. En effet, une fois que le plan d'action est réalisé, il est déployé sur un système d'information de médiation (SIM). Le SIM est capable d'interconnecter les différents systèmes d'informations des organisations mobilisées afin de supporter leur interopérabilité technologique. L'exécution du plan est ensuite monitorée. Si une divergence entre le plan d'action théorique et sa mise en œuvre dans le monde réel est détectée, celui-ci peut être remis en cause via la ré-exécution des deux étapes précédentes. L'élaboration d'un plan d'action collaboratif [109, 111, 119], son déploiement au sein d'un système d'information de médiation [14, 136, 149] et le monitoring de son exécution [12, 101] constituent les trois étapes de la démarche MISE (cf. figure 4.2). L'approche retenue par le projet MISE est pleinement compatible avec le modèle conceptuel de la planification présenté dans le chapitre 1 et peut être considérée comme une démarche de planification dynamique.

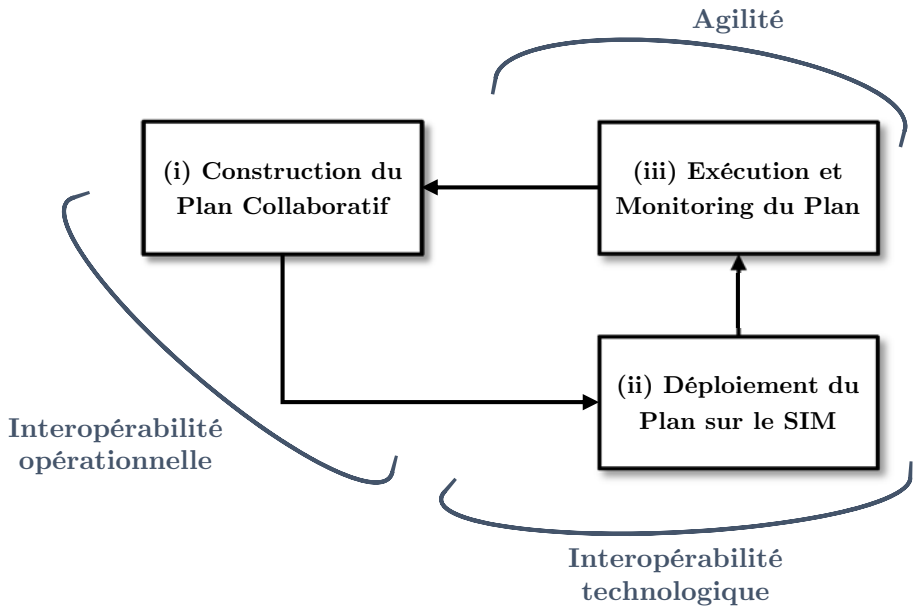


FIGURE 4.2 – Thèmes du projet de recherche MISE

4.2 De la construction de plans d'action collaboratifs

Un système capable de supporter des décideurs lors de la réalisation de plans d'action de réponse à une crise est introduit dans la section 4.2.1. Ce dernier repose sur des mécanismes de modélisation ainsi que des mécanismes de planification et d'aide à la décision (cf. sections 4.2.2 et 4.2.3).

4.2.1 Réalisation d'un système d'aide à la décision

Un système capable de proposer des plans collaboratifs pour supporter la coopération d'un ensemble d'entités organisationnelles ne peut être qu'un système d'aide à la décision. Il ne doit pas prétendre avoir pour vocation de se substituer au pouvoir du (ou des) décideur(s) mais vient au contraire épauler ce(s) dernier(s) lors de la prise de décision. En effet, l'exploitation faite des plans construits doit dépendre entièrement de la volonté du décideur. Il doit pouvoir les utiliser sans modification, les remanier avant de les déployer voire les refuser et demander au système d'en déduire de nouveaux (via la modification du problème de collaboration à résoudre).

Le système proposé a pour objectif d'aider les décideurs lors de la phase de réponse à la crise. Il convient de préciser que d'autres types de systèmes d'aide à la décision peuvent également être considérés pour supporter les membres de la cellule de crise tout au long du management de la crise (voir [143] pour plus de détails).

De par sa nature d'outil d'aide à la décision, un tel système est destiné à être utilisé par des experts opérationnels. Ceci impose naturellement plusieurs contraintes quant à sa conception et à son utilisation. Par exemple, pour que le système soit utilisable rapidement et sans difficulté, il faut que les utilisateurs puissent modéliser la situation collaborative (similaire à une *Common Operational Picture*) en utilisant des termes issus de leur langage opérationnel. En effet, il n'est pas concevable d'imaginer demander à ces derniers d'apprendre et de maîtriser un formalisme complexe. En outre, pour que les décideurs puissent réaliser des choix éclairés, il est nécessaire de leur fournir des éléments d'explication relatifs aux plans qui leurs sont suggérés. Ils peuvent ainsi mener une analyse comparative des différents plans collaboratifs construits par l'outil et comprendre avec précision pourquoi ces plans ont été proposés. Les décideurs maîtrisent alors pleinement la construction des plans collaboratifs. Ayant introduit ces deux prérequis, il est à présent possible de présenter un concept d'un système de construction de plans d'action collaboratifs (cf. figure 4.3).

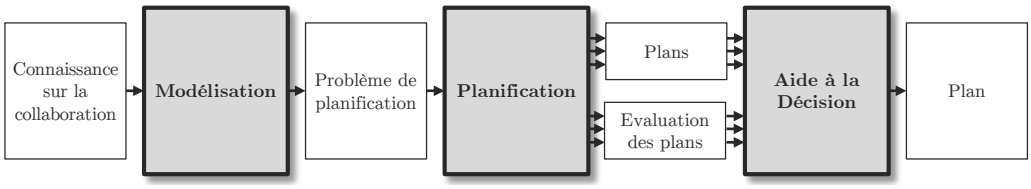


FIGURE 4.3 – Système de construction de plans d'action collaboratifs

Le système proposé s'utilise par l'invocation successive de trois étapes. L'étape de *modélisation* permet de représenter le problème collaboratif à résoudre en langage PDDL. Sous cette forme, il peut être exploité lors de l'étape de *planification* afin de produire un ou plusieurs plans d'action collaboratifs ainsi que leurs évaluations respectives. Finalement, une étape d'*aide à la décision* permet aux utilisateurs de choisir le plan qu'ils souhaitent exécuter. Les mécanismes relatifs aux différentes étapes de cette démarche de construction sont précisés ci-dessous avant d'être présentés en détails dans les sections 4.2.2 et 4.2.3.

Etape de modélisation du problème collaboratif

L'étape de modélisation a pour but de décrire la situation de crise, les partenaires mobilisés pour la résoudre et les objectifs et préférences des membres de la cellule de crise. La formalisation de cette connaissance est réalisée dans le langage opérationnel des utilisateurs avant d'être transformée automatiquement en PDDL (cf. figure 4.4).

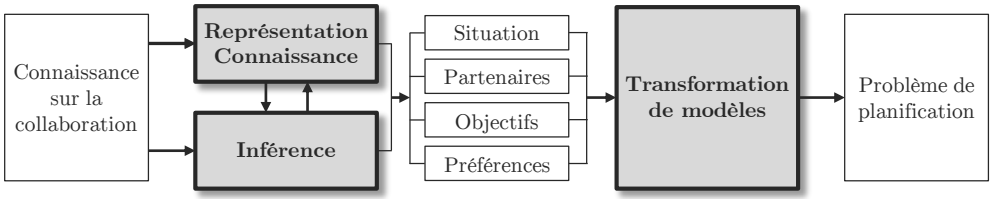


FIGURE 4.4 – Etape de modélisation du problème collaboratif

Lors de la représentation du problème, les membres de la cellule de crise réalisent quatre modèles (*situation*, *partenaires*, *objectifs* et *préférences*) à partir des informations dont ils disposent. Sur la base de situations de référence passées ou de doctrines établies à froid, des mécanismes d'inférence (déduction logique) peuvent être mis en œuvre afin d'aider les utilisateurs dans leur représentation du problème (voire qu'ils n'aient plus qu'à valider des modèles pré-remplis par le système comme suggéré dans [128]). Il convient de noter que les choix de modélisation effectués constituent un acte de prise de décision puisque les plans collaboratifs sont directement construits à partir des modèles réalisés.

Le *modèle de situation* contient les éléments du monde qu'il est pertinent de considérer du point de vue des décideurs dans le contexte de la collaboration étudiée. Il permet d'établir une description de la situation initiale de la crise. De plus, chaque entité organisationnelle impliquée dans la collaboration alimente un *modèle de partenaire* qui précise les capacités qu'elle peut mettre en œuvre ainsi que les ressources potentiellement partageables dont elle dispose. Le *modèle d'objectifs* décrit quant à lui l'ensemble des objectifs que les membres de la cellule de crise ont décidé d'accomplir. Finalement, les préférences des décideurs sont capturées à travers le *modèle de préférences*. Ce dernier permet d'évaluer et de comparer les différents plans solutions les uns par rapport aux autres. Plus la note d'un plan est élevée au regard du modèle de préférences et plus celui-ci maximise la satisfaction des décideurs.

Ces quatre modèles constitués (à l'aide d'éléments de langage opérationnel), une transformation de modèles [89] est appliquée afin de générer un problème de planification formalisé en langage PDDL.

Etapes de planification et d'aide à la décision

Les mécanismes de construction et d'évaluation des plans collaboratifs reposent sur la séquence de trois étapes introduite sur la figure 4.5. Dans un premier temps, le planificateur CHOPLAN est utilisé pour construire des plans d'action collaboratifs. Il convient de remarquer que la construction de plans collaboratifs constitue bien un problème de planification avec préférences. En effet, l'objectif des décideurs n'est pas de trouver une solution quelconque au problème mais un « bon » plan d'un point de vue opérationnel.

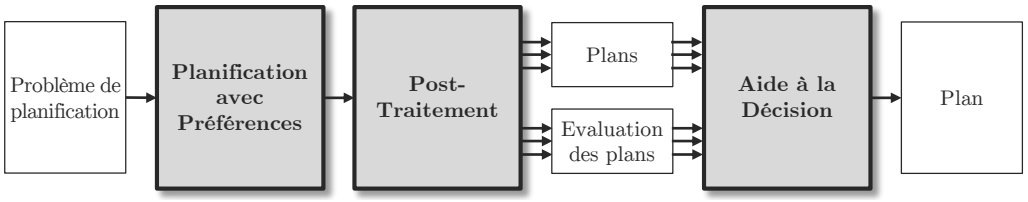


FIGURE 4.5 – Etapes de planification et d'aide à la décision

Le planificateur CHOPLAN est utilisé pour construire un plan d'action collaboratif pour chaque modèle de préférences des décideurs. Les plans ainsi générés sont optimisés au regard du modèle de préférences auxquels ils répondent et évalués selon tous les autres modèles de préférences disponibles. Pour un problème de collaboration donné, si les décideurs spécifient trois modèles de préférences M_1, M_2, M_3, trois plans d'action collaboratifs P_1, P_2, P_3 seront construits ainsi que les neufs évaluations correspondantes ($P_1 \leftrightarrow M_1$, $P_1 \leftrightarrow M_2$, $P_1 \leftrightarrow M_3$, $P_2 \leftrightarrow M_1 \cdots$). Dans ce cas, le plan P_1 (respectivement P_2 et P_3) maximise la satisfaction des décideurs correspondant au modèle de préférences M_1 (respectivement M_2 et M_3).

Lors de l'étape de post-traitement, la séquence d'action construite par CHOPLAN est transformée afin de paralléliser au maximum les différentes actions du plan tout en s'assurant que ce dernier reste réalisable. Cette parallélisation permet généralement de diminuer le temps d'exécution du plan ce qui est crucial dans la plupart des cas de gestion de situations de crise.

Finalement, l'étape d'aide à la décision permet aux décideurs de visualiser et d'analyser les plans d'action collaboratifs suggérés par le système afin de choisir celui qu'ils souhaitent déployer. Pour cela, l'outil expose des éléments d'analyse relatifs aux différents plans et modèles de préférences considérés. En conséquence, les décideurs peuvent comparer les plans les uns avec les autres avant d'effectuer leur choix.

Architecture et implémentation du système

Afin de démontrer la faisabilité du système décrit dans cette section, un prototype a été développé. Ce dernier s'appuie sur une architecture orientée services [54] ce qui lui permet notamment d'être compatible avec le système d'information de médiation utilisé dans la démarche MISE (voir [13] pour plus de détails). Les différents éléments présentés sur les figures 4.4 et 4.5 sont par conséquent implémentés à l'aide de services web ce qui confère au prototype une grande modularité.

En outre, le prototype peut être utilisé à partir d'un navigateur web ce qui lui permet d'être rapidement déployé lors de n'importe quelle situation réelle. La partie cliente du prototype a été implémentée avec les technologies web classiques (HTML, CSS, JS) tandis que la partie serveur a été développée à l'aide du langage Java. La figure 4.6 précise les modules du système qui ont effectivement été réalisés dans le cadre de cette étude.

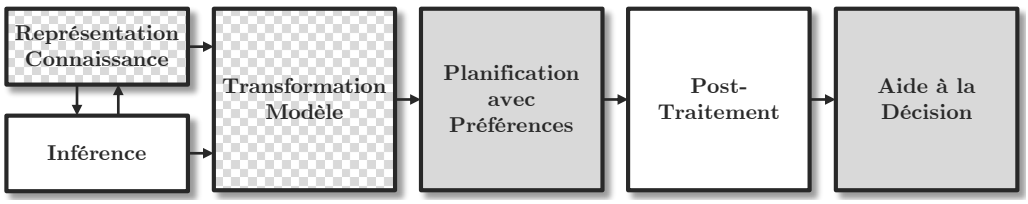


Figure 4.6 – Etat d'avancement du prototype. Les éléments sur fond blanc n'ont pas été implémentés tandis que les éléments sur fond gris (respectivement hachuré) sont entièrement (respectivement partiellement) implémentés.

4.2.2 Modélisation du problème collaboratif

Cette section détaille l'étape de modélisation qui permet à des décideurs de représenter un problème de collaboration au sein d'un aréopage d'entités organisationnelles (cf. figure 4.4). Dans un premier temps, des mécanismes de représentation et raisonnement sur les connaissances sont mis en œuvre afin d'aider les décideurs à réaliser un modèle de la situation, un modèle des partenaires mobilisés, un modèle des objectifs à atteindre ainsi qu'un modèle de leurs préférences. Une transformation de modèles est ensuite utilisée afin de générer le problème de planification correspondant.

Représentation des connaissances et inférence

L'activité de métamodélisation est la clef de voute de la représentation des connaissances. Un *métamodèle* est un ensemble de concepts et de règles relatives à ces derniers qui sont utilisés pour construire des modèles. Par conséquent, un métamodèle impose une structure quant à la représentation des connaissances considérées. Ceci permet de guider les utilisateurs lors de la création des modèles tout en s'assurant que ces derniers auront la cohérence nécessaire pour être manipulés par des outils de traitement automatiques.

Plusieurs métamodèles pour caractériser l'interopérabilité des organisations ont été proposés dans le cadre du projet de recherche MISE à l'image de celui dédié à la collaboration d'entreprises [119] ou encore celui pour la gestion collaborative de situations de crise [16, 138]. Par la suite, un *métamodèle cœur* qui généralise ces travaux a également été proposé [102]. Le métamodèle présenté dans cette section (cf. figure 4.7) se limite aux concepts effectivement utilisés dans cette étude.

Le *métamodèle de collaboration* proposé est organisé selon plusieurs modules qui s'articulent les uns par rapport aux autres [22]. Le *module cœur* (qui est utilisé pour construire les modèles de partenaires et d'objectifs) formalise l'essence d'une collaboration. Plusieurs *partenaires* qui fournissent des *capacités* et des *ressources* se regroupent pour former un *réseau collaboratif* (à savoir la cellule de crise) afin d'atteindre un certain nombre d'*objectifs*. Il convient de remarquer que l'existence d'une relation d'agrégation sur le concept de *capacité* permet aux partenaires

de la gestion de crise d'exposer leurs capacités avec la granularité de leur choix. Ces derniers peuvent ainsi décrire leurs savoir-faire avec un niveau de détails cohérent avec le type d'interventions pour lesquelles ils acceptent d'être sollicités. Le *module de situation* regroupe l'ensemble des concepts nécessaires pour décrire l'*environnement* de la collaboration. Les *composants d'environnement* qui peuvent être caractérisés par divers *états* représentent l'ensemble des éléments du monde qu'il est pertinent de considérer dans le cadre de la collaboration. Pour finir, le *module de préférences* formalise, comme son nom le suggère, les *préférences* des membres de la cellule de crise ; lesquelles peuvent être agrégées entre elles pour constituer un *modèle de préférences*.

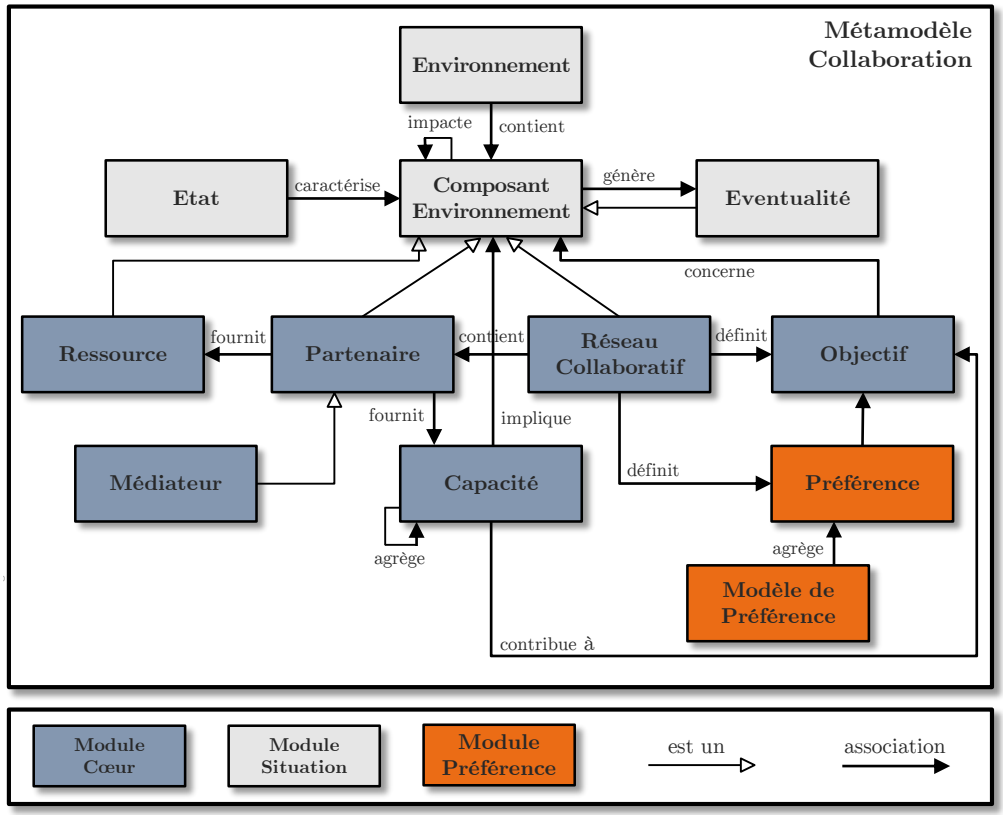


FIGURE 4.7 – Métamodèle de collaboration

Un métamodèle peut être intrinsèquement considéré comme un modèle de son domaine d'intérêt (par exemple, le métamodèle de la figure 4.7 constitue une modélisation du domaine de l'interopérabilité des organisations). A ce titre, un métamodèle décrit la structure d'une *ontologie*, concept défini comme « une spécification explicite d'une conceptualisation » [75]. Cela signifie que des instances des modèles réalisés à l'aide du métamodèle présenté sur la figure 4.7 peuvent être injectées dans une ontologie afin de construire une base de connaissances pour la collaboration d'entités organisationnelles. En conséquence, un système expert peut être utilisé pour appliquer des mécanismes d'inférence sur cette base de connaissances afin de supporter les décideurs lors de la modélisation du problème. Des déductions logiques peuvent ainsi être utilisées pour aider les utilisateurs à construire leurs modèles plus rapidement, suggérer des éléments de modélisation aux décideurs ou encore pour réaliser une réconciliation sémantique entre les différents concepts modélisés par les utilisateurs.

A titre d'exemple, si l'ontologie contient une description des capacités de la BSPP (Brigade de Sapeurs-Pompiers de Paris) et qu'un SDIS (Service Départemental d'Incendie et de Secours) fait partie des partenaires mobilisés, alors il est possible de déduire que certaines des capacités de la BSSP sont potentiellement des capacités d'un SDIS. Ce type de déduction aussi triviale soit-elle permet de pré-remplir le modèle du partenaire SDIS lui facilitant ainsi la tâche tout en lui faisant gagner du temps. De plus, en comparant le modèle de situation réalisé par les décideurs à des situations de référence passées, il peut être possible de suggérer certains objectifs à ces derniers. Le système peut par exemple proposer aux utilisateurs de considérer un risque particulier sur la base de retours d'expérience. En outre, les modèles étant réalisés par plusieurs personnes distinctes, il se peut que plusieurs termes aient été employés pour décrire le même concept (« feu » et « incendie » par exemple). Si de tels cas n'entravent généralement pas la réflexion humaine, ils peuvent néanmoins mettre en péril les traitements et raisonnements automatiques effectués par des machines. Il est donc nécessaire d'effectuer une *réconciliation sémantique* entre les différents modèles réalisés afin de supprimer d'éventuelles ambiguïtés. Ceci peut être facilité par la mise en oeuvre de langages pivots ou par l'utilisation de règles d'inférence sur une ontologie puisque cette dernière contient par nature un grand nombre des concepts du domaine considéré (voir [139] par exemple).

Transformation de modèles

Une démarche de transformation de modèles s'appuie sur des connaissances contenues dans un *modèle source* afin de construire (éventuellement automatiquement) un *modèle cible*. La notion de métamodèle est au cœur du processus de transformation de modèles comme illustré sur la figure 4.8. En effet, les concepts partagés par le *métamodèle source* et le *métamodèle cible* définissent les parties spécifiques et partagées des deux modèles considérés. Ainsi, il est possible d'extraire la connaissance de la partie partagée du modèle source puis de la transformer via des *règles de mapping* afin de produire la connaissance qui caractérise la partie partagée du modèle cible. De plus, la connaissance spécifique du modèle source est capitalisée pour une éventuelle utilisation ultérieure tandis que la connaissance spécifique du modèle cible doit être ajoutée afin de compléter ce dernier. Pour que la transformation soit réalisable, les modèles considérés doivent respecter la structure imposée par leurs métamodèles respectifs ; lesquels doivent en outre nécessairement posséder un ensemble de concepts communs.

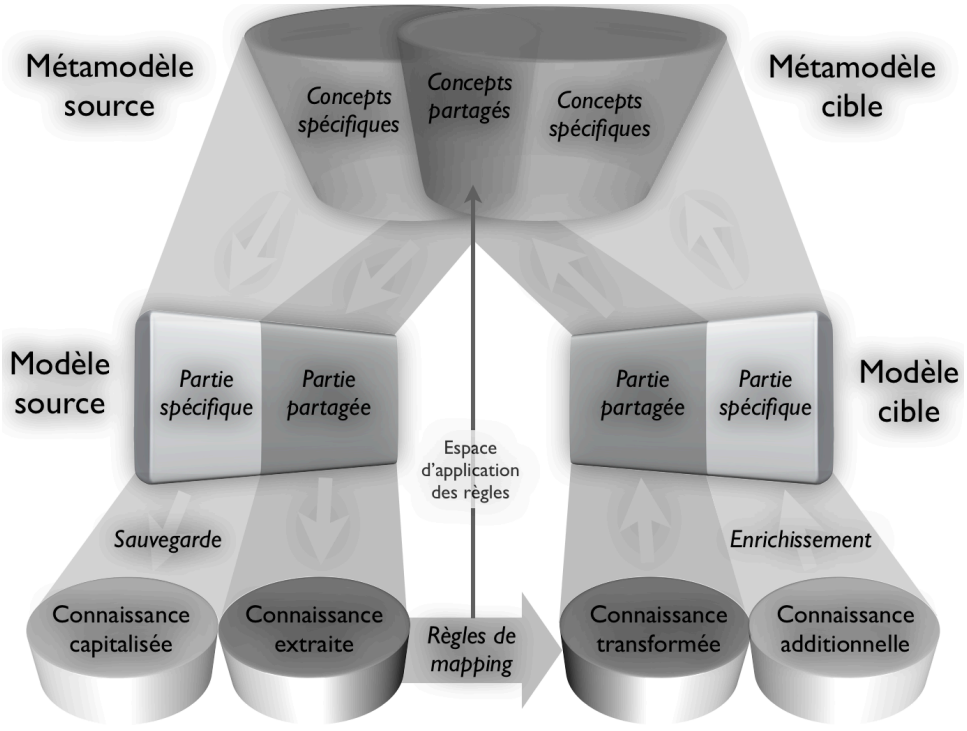


Figure 4.8 – Principe de la transformation de modèles ; d'après [13, 140]

Pour réaliser la transformation de modèles considérée dans cette étude, les modèles de situation, de partenaires, d'objectifs et de préférences sont dans un premier temps fusionnés afin de créer un modèle de collaboration. Ce dernier constitue le modèle source de la transformation à partir duquel il faut générer un problème de planification. Afin d'expliquer conceptuellement comment le problème opérationnel est transformé en problème de planification, un métamodèle de la planification est introduit à l'aide de la figure 4.9. Ce métamodèle peut être considéré comme une abstraction de la syntaxe BNF du langage PDDL [63].

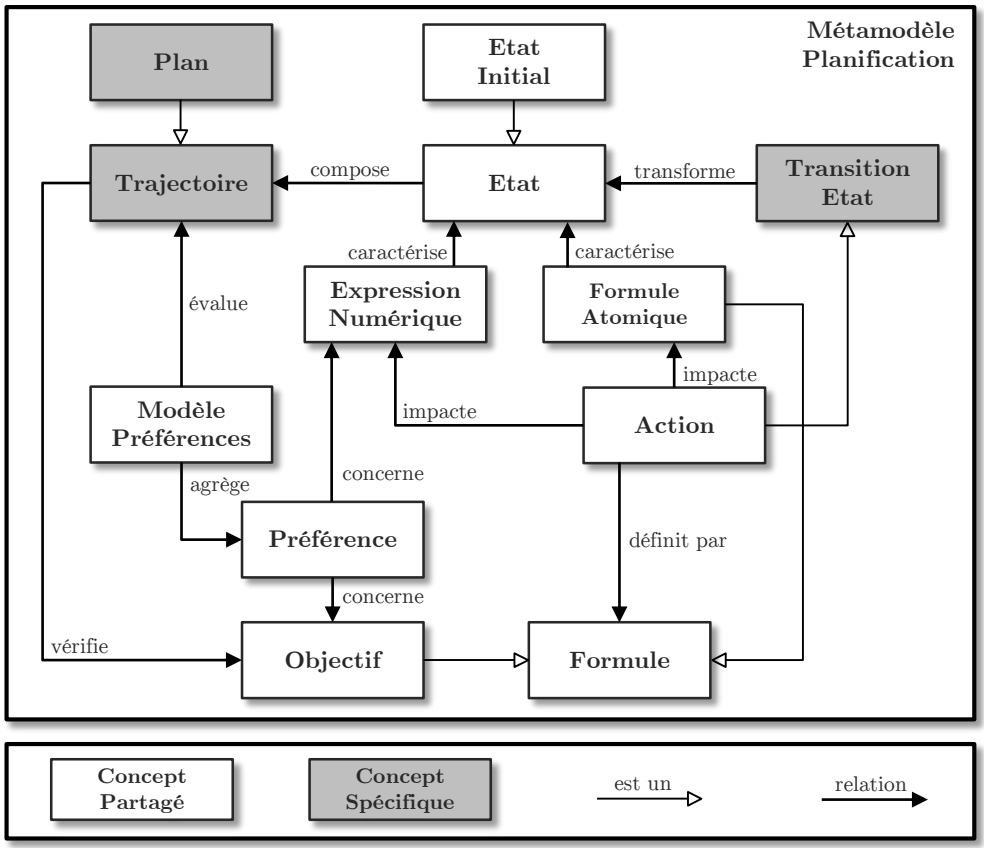


FIGURE 4.9 – Métamodèle de planification. Les concepts communs avec le métamodèle de collaboration sont représentés sur fond blanc tandis que les concepts spécifiques à ce métamodèle sont représentés sur fond gris.

Le tableau 4.1 précise les concepts partagés par les deux métamodèles présentés précédemment (cf. figures 4.7 et 4.9). Ce dernier capture par conséquent l'essence de la transformation de modèles réalisée. Ainsi, les règles de mapping peuvent être obtenues en déclinant techniquement les correspondances présentées dans ce tableau (transformation d'une syntaxe XML/JSON en langage PDDL). Par ailleurs, les concepts de *réseau collaboratif*, *partenaire* et *médiateur* constituent la connaissance spécifique du modèle source. En effet, les actions n'étant pas différenciées en fonction de l'acteur qui les exécute dans le langage PDDL, ces concepts n'ont pas d'équivalent dans le modèle de planification. Cette connaissance est donc capitalisée lors de la transformation de modèles puis utilisée ultérieurement pour réaffecter les actions des plans solutions à leurs acteurs respectifs. Ce mécanisme permet de transformer les plans générés par ChoPlan en véritable plans d'action collaboratifs au sens de la définition 4.1. Par ailleurs, les concepts spécifiques au modèle cible sont sans surprise les éléments utilisés pour définir la notion de plan (*transition d'état*, *trajectoire* et *plan*). En effet, ces éléments n'ont pas d'équivalent dans le modèle de collaboration puisque ce dernier a uniquement pour vocation de supporter la spécification de la situation initiale, des partenaires mobilisés, des objectifs à atteindre et des préférences à optimiser.

Métamodèle Collaboration	**Métamodèle Planification**
Capacité	Action
Composant d'Environnement	Formule Atomique
Environnement	Etat Initial
Etat	Formule Atomique
Eventualité	Formule Atomique
Modèle de Préférences	Modèle de Préférences
Objectif	Objectif
Préférence	Préférence
Ressource	Expression Numérique

Tableau 4.1 – Correspondances entre les concepts partagés par les métamodèles de collaboration (source) et de planification (cible)

4.2.3 Construction et évaluation de plans collaboratifs

Les mécanismes présentés dans la section 4.2.2 permettent de construire un problème de planification qui correspond à la problématique que les membres de la cellule de crise ont décidé d'adresser. Cette section détaille les trois étapes de résolution de ce problème de planification (cf. figure 4.5) à savoir (i) la construction d'un ou plusieurs plans, (ii) le post-traitement de ces derniers puis (iii) le support apporté aux décideurs lors du choix du plan à exécuter.

Planification avec préférences

Pour des cas d'application réels, la question de la confiance des décideurs vis-à-vis du système est cruciale. En effet, pour que ces derniers s'approprient (et donc in fine mettent en œuvre) les plans proposés, il est nécessaire qu'ils comprennent parfaitement pourquoi ces plans leurs ont été suggérés. En conséquence, la notion de préférences est fondamentale puisqu'elle permet aux décideurs de spécifier avec précision leurs exigences par rapport à la solution attendue. Ce constat motive la mise en œuvre d'une démarche de planification avec préférences pour résoudre les problèmes de gestion de crise. De plus, l'utilisation de ChoPlan semble particulièrement pertinente puisque ce dernier offre un grand pouvoir expressif aux décideurs quant à la représentation de leurs préférences comme expliqué dans le chapitre 3.

ChoPlan est utilisé pour construire un plan pour chaque modèle de préférences fourni par les membres de la cellule de crise. L'utilisation de plusieurs modèles de préférences permet aux décideurs de considérer plusieurs stratégies de résolution différentes. Ceci peut être particulièrement utile dans le cas où plusieurs membres de la cellule de crise ne sont pas d'accord quant à l'importance relative à associer aux différentes préférences considérées. En effet, il est alors possible de résoudre une instance du problème de planification pour chaque modèle de préférences réalisé (un par décideur par exemple) puis de comparer les solutions obtenues entre elles. Une autre approche permettant de résoudre cette problématique consiste à agréger l'ensemble des opinions des décideurs en un unique modèle de préférences comme proposé dans [84].

Les mécanismes de configuration de ChoPlan étant relativement souples, les décideurs peuvent adapter le temps alloué à chaque résolution en fonction de leurs besoins opérationnels. En outre, les problématiques d'interopérabilité entre organisations sont d'une granularité telle que les problèmes de planification correspondants sont souvent moins combinatoires que ceux considérés lors des compétitions de planification. Ainsi, ChoPlan peut être en mesure d'identifier des solutions satisfaisantes même lorsque les contraintes opérationnelles imposent un faible temps de recherche (de l'ordre de la minute par exemple).

De plus, comme mentionné dans la section 4.2.1, l'architecture de ChoPlan est relativement ouverte ce qui permet d'ajouter facilement de nouvelles règles de sélection ou règles de coupe à l'outil. En particulier, il est possible d'ajouter des règles opérationnelles pour privilégier certains nœuds à d'autres au cours de la recherche. Ces règles peuvent capturer des éléments qui affectent la prise de décision sans pour autant être formalisés au sein du problème à l'image de retours d'expérience opérationnels par exemple. Un tel ajout fait perdre à ChoPlan sa portée de solveur générique puisque ces règles ne sont valides que pour un domaine particulier mais peut lui permettre d'être plus pertinent lors de la résolution des problèmes de ce domaine.

Post-traitement des plans

ChoPlan respecte l'hypothèse H5 dite des plans séquentiels du modèle conceptuel de la planification (cf. section 1.1.1) et génère par conséquent des séquences d'actions linéaires. L'étape de post-traitement a pour but de paralléliser les différentes actions des plans lorsque cela est possible ce qui permet généralement de diminuer le temps d'exécution réel de ces derniers.

La parallélisation des plans peut être réalisée à l'aide de la notion d'*actions mutuellement exclusives* (ou *actions mutex*) [61]. Afin de définir précisément cette notion, la définition d'une action close a (cf. section 1.1.3) est enrichie à l'aide des trois ensembles suivants :

- L_a : l'ensemble des expressions numériques qui apparaissent dans une *lvalue* de a ;

- R_a : l'ensemble des expressions numériques qui apparaissent dans une *rvalue* de a ou dans une précondition de a ;
- L_a^* : l'ensemble des expressions numériques qui apparaissent dans une *lvalue* d'un effet numérique additif de a.

Intuitivement, deux actions sont mutuellement exclusives lorsqu'elles interfèrent l'une avec l'autre. Ainsi, pour que deux actions a et b puissent être exécutées en parallèle, elles doivent être non mutex (voir définition 4.2).

Définition 4.2 - Actions non mutex [61]
Deux actions a et b sont non mutuellement exclusives si elles vérifient les quatre conditions suivantes :

$$GPre_a \cap (Add_b \cup Del_b) = GPre_b \cap (Add_a \cup Del_a) = \emptyset$$
$$Add_a \cap Del_b = Add_b \cap Del_a = \emptyset$$
$$L_a \cap R_b = R_a \cap L_b = \emptyset$$
$$L_a \cap L_b \subseteq L_a^* \cup L_b^*$$

La première condition impose que les préconditions de l'action a ne soient pas affectées par les effets de b (et vice-versa). La deuxième condition s'assure quant à elle que les effets de a et b restent cohérents lorsque les deux actions sont exécutées simultanément. La troisième condition impose que les valeurs affectées par les effets numériques de a ne soient utilisés ni dans les préconditions de b ni pour calculer l'un des effets numériques de b (et inversement). Finalement, la quatrième condition précise que deux actions concurrentes ne peuvent affecter la même valeur que si elles le font toutes deux par l'intermédiaire d'effets numériques additifs.

Un ensemble d'actions A_H est un *ensemble d'actions non mutuellement exclusives* si $\forall a, b \in A_H$, a et b sont non mutex [61]. Par définition, toutes les actions de A_H peuvent être parallélisées puisqu'elles sont toutes exécutables simultanément. En conséquence, il est possible de paralléliser une séquence d'actions linéaire (un plan) en la transformant en séquence d'ensemble d'actions non mutex. Une telle transformation peut être réalisée à l'aide de l'algorithme 4.1. Dans ce cas, il convient alors de redéfinir les notions de structure, d'exécution et de validité d'un plan (voir [61] pour une formalisation complète).

Cette étape de post-traitement peut être réalisée a posteriori de la construction des plans et ce sans qu'aucune modification ne soit apportée au planificateur. Pour conduire des raisonnements plus poussés quant à la concurrence des actions du plan, il serait nécessaire d'intégrer de véritables mécanismes d'ordonnancement dans CHOPLAN. Il faudrait notamment étendre ce dernier pour qu'il gère les aspects temporels des problèmes de planification (durée explicite des actions...).

```
Input :  lplan séquence d'actions
Output : pplan séquence d'ensembles d'actions non mutex
Data :   action action
         set ensemble d'actions non mutex

Algorithm PARALLELIZE()
    set ← newList()
    for i = 1 to lplan.size() do
        action ← lplan.get(i)
        if ALLNONMUTEX(action, set) = true then
            set.add(action)
        else
            pplan.add(set)
            set ← newList()
            set.add(action)
        end
    end
    return pplan

Function ALLNONMUTEX(action, set)
    for i = 1 to set.size() do
        if MUTEX(action, set.get(i)) = true then
            return false
        end
    end
    return true
```

Algorithme 4.1 : Post-traitement des plans. La fonction `MUTEX` retourne vrai lorsque les deux actions passées en paramètres sont mutex.

Aide à la décision

Lors de l'étape d'aide à la décision, une interface permettant de visualiser rapidement l'ensemble des résultats produits par CHOPLAN est mise à disposition des décideurs. Cette interface (voir section 4.3.3 pour plus de précisions) permet de visualiser les différents plans solutions et possède un tableau qui récapitule les notes des plans par rapport à chaque modèle de préférences considéré. Ainsi, si les décideurs ont défini deux modèles de préférences P_1 et P_2, deux solutions S_1 et S_2 (respectivement optimisées par rapport à P_1 et P_2) sont construites. Dans ce cas, le tableau récapitulatif comporte quatre valeurs à savoir les évaluations $P_1 \leftrightarrow S_1$, $P_1 \leftrightarrow S_2$, $P_2 \leftrightarrow S_1$ et $P_2 \leftrightarrow S_2$.

De plus, deux graphiques explicatifs sont construits pour chaque évaluation (c.-à-d. pour chaque couple plan/modèle de préférences) comme illustré sur la figure 4.10. Le premier graphique précise les valeurs obtenues par le plan pour chacun des critères du modèle de préférences. Le second graphique utilise la représentation graphique de la valeur d'une intégrale de Choquet proposée dans [95]. Il aide les décideurs à comprendre en détails les notes obtenues par les différents plans. La propriété de monotonicité des capacités et la définition des indices d'interaction entre critères imposent :

$$\forall i, j \in P, \Big[\phi_i - \frac{1}{2} \sum_{j \neq i} |I_{ij}| \Big] \geq 0 \tag{4.1}$$

$$\sum_{I_{ij}>0} I_{ij} + \sum_{I_{ij}<0} |I_{ij}| + \sum_{i \in P} \Big[\phi_i - \frac{1}{2} \sum_{j \neq i} |I_{ij}| \Big] = 1 \tag{4.2}$$

Tous les coefficients d'une intégrale de Choquet 2-additive sont donc non négatifs et leur somme vaut 1 (cf. définition 2.11). Ainsi, cette dernière peut s'écrire comme une somme $C_\mu(x) = \sum_k \alpha_k \, c_k(x)$ dont les éléments c_k représentent des critères seuls, des conjonctions de critères (synergie positive entre i et j) ou des disjonctions de critères (synergie négative entre i et j). En conséquence, la valeur d'une intégrale de Choquet 2-additive peut astucieusement être représentée par un diagramme en secteurs. En effet, il est possible d'associer à chaque élément c_k un secteur circulaire de longueur d'arc $2\pi\alpha_k$. De plus, si tous ces secteurs circulaires ont un taux de remplissage de $c_k(x)$ alors le taux de remplissage du diagramme en secteurs représente la valeur $C_\mu(x)$. L'interprétation de ce graphique est alors relativement

intuitive puisque plus la surface colorée du diagramme en secteurs est grande et plus la note du plan est élevée (cf. figure 4.10).

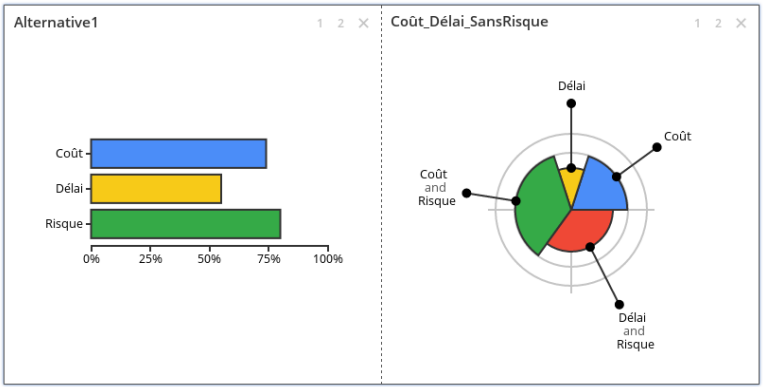


FIGURE 4.10 – Evaluation d'un plan par rapport à un modèle de préférence

En outre, les décideurs ont également la possibilité d'éditer les plans proposés par l'outil. Ceci leur permet de se les approprier plus facilement et d'y apporter des modifications si nécessaire (la validité du plan ainsi modifié n'étant néanmoins pas assurée par l'outil). Pour finir, il convient de rappeler que si aucun des plans proposés n'est satisfaisant, les décideurs peuvent modifier les modèles qu'ils ont réalisés puis demander à l'outil de résoudre le problème à nouveau.

4.3 Exemple de résolution d'une situation de crise

Afin d'illustrer le fonctionnement du prototype développé dans le cadre de ces travaux, un cas d'étude issu du domaine de la gestion de crise est présenté. Cet exemple illustratif est détaillé dans la section 4.3.1. La section 4.3.2 s'intéresse à la modélisation du problème opérationnel tandis que la section 4.3.3 traite de sa résolution.

Le cas d'étude considéré est inspiré d'un scénario opérationnel utilisé dans le cadre du projet européen SECTOR [38]. Ce projet ambitionne de concevoir un *espace d'information partagé* (*Common Information Space* ou *CIS*) par les différents acteurs de la gestion de crise. Le CIS propose un certain nombre de services dits à valeur ajoutée qui permettent de faciliter la mission des acteurs de la gestion

de crise. Une version préliminaire des travaux présentés dans ce manuscrit a été intégrée à la base d'un service à valeur ajouté au sein du CIS du projet SECTOR. Ce service constitue un module d'aide à la planification collaborative qui permet aux décideurs de la gestion de crise d'élaborer des plans d'actions capables de supporter l'interopérabilité opérationnelle des différents partenaires mobilisés.

4.3.1 Description du scénario opérationnel

Le cas considéré a été retenu pour son réalisme puisqu'il a fait l'objet d'une définition et d'une validation de la part des partenaires opérationnels du projet SECTOR. Ce dernier met en œuvre un scénario de grande inondation en Europe du Nord. Plus précisément, il se déroule aux alentours de la ville de Roermond qui se situe dans la région de Limburg aux Pays-Bas. Le scénario se déroule lors d'un hiver au cours duquel de fortes précipitations ont été enregistrées pendant plusieurs mois. En conséquence, le volume des rivières a atteint son maximum. Par ailleurs, les prévisions météorologiques suggèrent que de fortes pluies sont encore à prévoir. La ville de Roermond et ses alentours sont particulièrement exposés en cas d'inondation puisque cette zone géographique est traversée par plusieurs rivières (la Meuse notamment) et abrite de nombreux plans d'eau naturels.

L'objectif de ce cas d'étude est de gérer au mieux la situation de crise qui serait provoquée par les inondations à venir. Dans le cadre du projet SECTOR, la gestion de cette situation critique mobilise des opérateurs de secours pendant plusieurs jours. Le module de planification collaborative est utilisée à plusieurs reprises lors de la résolution du problème via une mise à jour des modèles lorsque la situation évolue (ce qui constitue une mise en œuvre manuelle des mécanismes d'agilité de la démarche MISE). Cette section illustre la première utilisation du prototype à savoir l'élaboration du premier plan d'action collaboratif considéré par les décideurs.

L'interface du prototype est présentée sur la figure 4.11. Ce dernier est utilisé en mode « réponse », mode qui permet de construire des plans. Le menu vertical décrit les étapes d'utilisation du prototype : réalisation des modèles (« Modelling »), appel au planificateur ChoPlan (« Planning ») et aide à la décision (« Decision Making »). Dans la suite de cette section, l'interface du prototype n'est plus représentée.

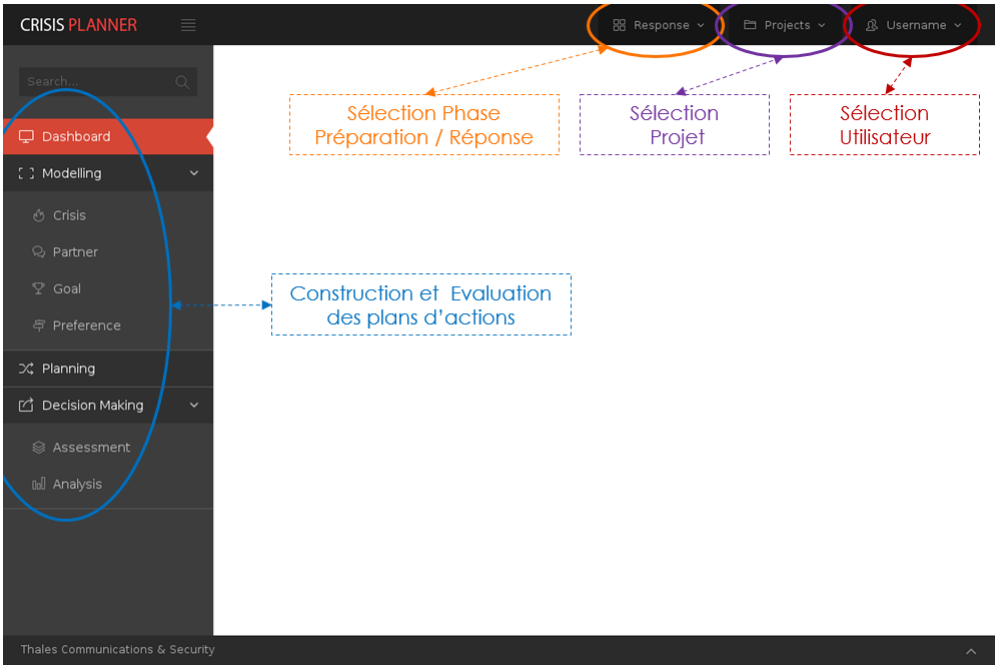


FIGURE 4.11 – Interface du prototype

4.3.2 Modélisation opérationnelle du problème

Cette section présente les quatre modèles réalisés (Situation, Partenaires, Objectifs et Préférences) pour représenter le problème collaboratif de ce cas d'étude. Le prototype réalisé ne permet de construire les modèles qu'à partir d'une unique machine. En supprimant cette contrainte technique, il serait possible de mettre en œuvre une démarche de modélisation collaborative ; laquelle permettrait à plusieurs utilisateurs, éventuellement distants, de participer en temps réel à l'élaboration des modèles. Il faudrait alors s'assurer que les éléments modélisés par les différents utilisateurs soient cohérents les uns avec les autres. Cette problématique est déjà adressée pour certains domaines industriels, voir par exemple [125].

Modélisation de la situation

Un modeleur graphique permet aux décideurs de décrire l'*environnement* de la collaboration. Pour cela, ces derniers peuvent employer les concepts de *composant d'environnement* (rectangle bleu à bords arrondis), d'*éventualité* (trapèze rectangle

violet) et d'*états* (ellipse verte) du métamodèle de collaboration présenté précédemment (cf. figure 4.7). Ces différents concepts peuvent être reliés les uns aux autres conformément aux relations du métamodèle. Ainsi, un *composant d'environnement* peut *générer* (flèche grise) une *éventualité* ou encore *impacter* (flèche violette) un autre *composant d'environnement*. En outre, les *composants d'environnement* peuvent être *caractérisés* (flèche verte) par des *états*. Le modeleur graphique permet aux décideurs de représenter la situation rapidement tout en leur interdisant de créer des relations non conformes à celles spécifiées dans le métamodèle de collaboration. Le modèle de situation du cas d'étude considéré est présenté sur la figure 4.12.

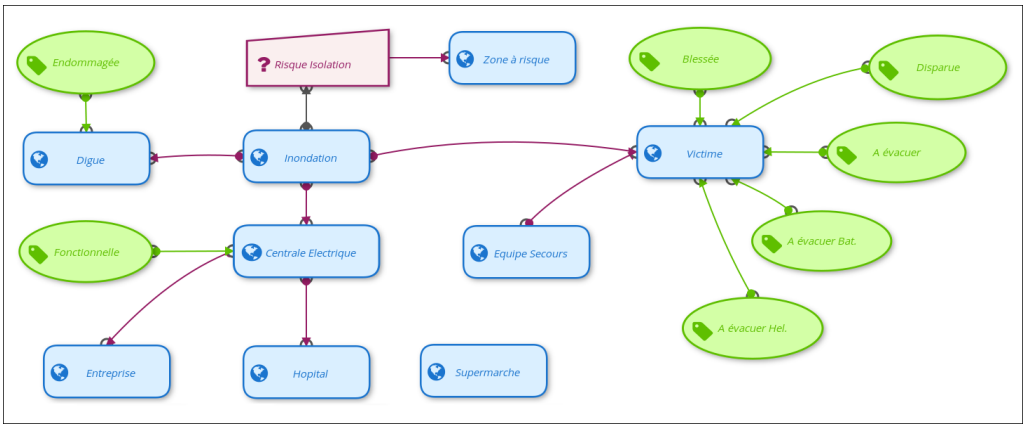


Figure 4.12 – Modèle de situation conforme au métamodèle de collaboration

Bien que l'inondation ait été anticipée, cette dernière est survenue plus rapidement que prévu à cause d'un fort événement climatique dont l'ampleur n'avait pas été correctement estimée. En conséquence, tous les dispositifs de prévention n'ont pas encore été déployés au début de la crise. Sans surprise, le composant d'environnement central de ce modèle est l'*inondation* à traiter. Cette dernière a *endommagé* une *digue* et impacte une *centrale électrique* qui alimente actuellement l'*hôpital* et les usines de production d'une grande *entreprise* de la région. Par ailleurs, elle génère également un risque d'isolation pour les habitants d'une zone géographique surélevée qui bien qu'a priori non inondable pourrait devenir complètement inaccessible si le niveau d'eau continuait de croître. De plus, l'inondation a fait plusieurs victimes qui peuvent être blessées, disparues ou qui doivent être évacuées (par bateau ou hélicoptère dans les cas les plus difficiles). Pour finir, il convient de préciser que le modèle de la figure 4.12 a été simplifié pour des fins d'illustrations. Pour représenter

pleinement le problème, il aurait fallu créer autant de composants d'environnement *Victime* et *Equipe Secours* qu'il y a de victimes et d'équipes de secours dans le scénario. Ceci s'explique par le choix des décideurs de modéliser ces éléments comme des composants d'environnement et non pas comme des ressources (auquel cas un seul composant d'environnement associé à une quantité aurait été nécessaire).

Modélisation des partenaires

Le prototype fournit au décideur une interface qui lui permet de visualiser l'ensemble des partenaires mobilisés pour résoudre la crise (cf. figure 4.13). Il est ensuite demandé à chaque partenaire de décrire l'ensemble des capacités qu'il peut mettre en œuvre (cf. colonne « Status »). Des profils de partenaires prédéfinis (appelés « Role » sur la figure 4.13) peuvent être utilisés pour aider les utilisateurs à modéliser leurs capacités plus rapidement. La figure 4.14 présente les capacités et ressources des pompiers néerlandais. Dans le cas d'étude considéré, la majorité des partenaires mobilisés sont des équipes de secours néerlandaises ou des acteurs locaux. De plus, une aide internationale peut être mobilisée pour renforcer les équipes de police et les équipes de la Croix Rouge si nécessaire.

Partner	Role	Edit	Status
Ministre de l'Intérieur - Pays-Bas	None selected		✓
Pompiers - Pays-Bas	Pompier		✓
Police - Pays-Bas	Force Ordre		✓
Croix Rouge - Pays-Bas	None selected		✓
Hopital Roermond	None selected		✓
Opérateur Electricité	None selected		✓
Supermarché Roermond	None selected		✓
Police - Allemagne	Force Ordre		✓
Croix Rouge - Allemagne	None selected		✗
Croix Rouge - Belgique	None selected		✗

Figure 4.13 – Liste des partenaires mobilisés dans le cas d'étude

	Partner	Capability	Requirement	Effect
☐	Pompiers - Pays-Bas	Eteindre incendie faible ampleur	Incendie [Faible ampleur] Equipe Secours Fourgon d'incendie [1]	Remove *Incendie [Faible ampleur]*
☐	Pompiers - Pays-Bas	Eteindre incendie grande ampleur	Incendie [Grande ampleur] Equipe Secours Fourgon d'incendie [2]	Remove *Incendie [Grande ampleur]*
☐	Pompiers - Pays-Bas	Réaliser évacuation	Victime [A évacuer] Equipe Secours	Modify *Victime [A évacuer]* to *Victime [Secourue]*
☐	Pompiers - Pays-Bas	Réaliser évacuation par bateau	Victime [A évacuer Bat.] Equipe Secours [Evacuation Bat.] Bateau sauvetage [1]	Modify *Victime [A évacuer Bat.]* to *Victime [Secourue]*
☐	Pompiers - Pays-Bas	Réaliser évacuation par hélicoptère	Victime [A évacuer Hel.] Equipe Secours [Evacuation Hel.] Helicoptère [1]	Modify *Victime [A évacuer Hel.]* to *Victime [Secourue]*
☐	Pompiers - Pays-Bas	Réaliser évacuation blessé	Victime [Blessé] Equipe Secours [PremierSoin] CentreSoin [AxePrioritaire]	Modify *Victime [Blessé]* to *Victime [Transféré CentreSoin]*
☐	Pompiers - Pays-Bas	Renforcer digue	Digue [Endommagée]	Modify *Digue [Endommagée]* to *Digue [Renforcée]*

	Partner	Resource	Quantity
☐	Pompiers - Pays-Bas	Couverture	120
☐	Pompiers - Pays-Bas	Fourgon d'incendie	10
☐	Pompiers - Pays-Bas	Bateau sauvetage	6

FIGURE 4.14 – Exemple de capacités et ressources dans le cas d'étude

Modélisation des objectifs

Le modeleur permettant de représenter les objectifs des décideurs n'a pas été développé dans le cadre de ces travaux. En conséquence, les objectifs du cas d'étude (représentés par le tableau 4.2) ont été ajoutés manuellement au problème PDDL lors de l'étape de transformation de modèles. Ces derniers concernent pour la plus grande partie du secours aux victimes ; lesquelles peuvent être blessées, avoir besoin d'être évacuées suite à la montée du niveau de l'eau ou encore être portées disparues. Par ailleurs, les membres de la cellule de crise ont décidé de placer l'hôpital de la ville en état d'alerte et d'installer un centre d'accueil provisoire pour les victimes. De plus, l'activité de la centrale électrique étant menacée par l'inondation, cette dernière doit être fermée. Il convient d'alimenter au préalable l'hôpital à l'aide de groupes électrogènes afin de s'assurer que ce dernier reste opérationnel. Des opérations de maintien de l'ordre (diffusion de consignes de sécurité, régulation de trafic...) sont également à prévoir afin de gérer le plus efficacement possible les axes routiers de la région (priorité aux secours et ravitaillement par exemple).

Objectifs
Add *centreAccueil* with state *[Operationnel]*
Modify *hopital* by adding state *[EtatAlerte]*
Modify *centraleElectrique* state from *[Fonctionnelle]* to *[Eteinte]*
Modify *hopital* by adding state *[AlimentéParGroupeElectrogène]*
Add *consigneSecuriteDiffusée* component
Add *traficRoutierRegulé* component
Modify *digue* state from *[Endommagée]* to *[Renforcée]*
Modify *victime01* state from *[A évacuer]* to *[Secourue]*
Modify *victime03* state from *[A évacuer Hel.]* to *[Secourue]*
Modify *victime04* state from *[Blessé]* to *[Soignée]*
Modify *victime13* state from *[A évacuer Bat.]* to *[Secourue]*
Modify *victime14* state from *[Disparue]* to *[Retrouvée]*
Modify *victime20* state from *[A évacuer]* to *[Secourue]*
...
hopital [AlimentéParGroupeElectrogène] Before *centraleElectrique [Etteinte]*

Tableau 4.2 – Exemple d'objectifs considérés dans le cas d'étude

Modélisation des préférences

Les modèles de préférence des décideurs sont réalisés à l'aide du logiciel Myriad (cf. figures 4.15, 4.16 et 4.17) qui a déjà été mentionné dans la section 2.4. Myriad permet aux décideurs de construire un modèle mathématique de leurs préférences sur la base d'informations préférentielles opérationnelles. Une fois que les modèles de préférence ont été élaborés, ces derniers peuvent être importés dans le prototype à l'aide d'une interface dédiée (cf. figure 4.18). Cette dernière permet également aux partenaires de préciser l'impact de leurs capacités sur les préférences numériques retenues.

Dans le scénario de cet exemple illustratif, les décideurs réalisent trois modèles de préférences qui correspondent aux trois stratégies de résolution qu'ils envisagent de mettre en œuvre (voir tableau 4.3). Le premier modèle de préférences évalue la performance de la réponse à la crise sur la base de trois critères : (i) l'*efficacité* des opérations de secours (qui a été modélisée par le nombre d'équipes de secours déployées) ; (ii) le *confort* des habitants de la zone soumise au risque d'isolation (qui varie s'il y a eu approvisionnement en nourriture et électricité pour les habitants ou non) et (iii) le *coût* de la réponse (hors coûts liés aux sauvetages des victimes). En outre, ce dernier interdit de recourir à l'aide internationale proposée par les nations voisines afin que les décideurs puissent apprécier la qualité de la solution qu'ils seraient en mesure de déployer avec les seules équipes de secours néerlandaises. Le deuxième modèle de préférences utilise les mêmes critères que le premier sans interdire l'utilisation de l'aide internationale. Ce dernier permettra ainsi d'estimer la plus-value apportée par l'intervention des équipes de secours allemandes et belges. Le troisième et dernier modèle de préférences comporte un critère supplémentaire qui concerne l'usine de production alimentées par la centrale électrique. En effet, l'arrêt total de la centrale électrique pourrait endommager l'usine ce qui entrainerait des pertes économiques pour la région à moyen terme. Aux trois préférences numériques *efficacité*, *confort* et *coût*, le troisième modèle ajoute donc une préférence de trajectoire dénotée *entreprise* qui traduit la volonté d'alimenter les usines de l'entreprise à l'aide de groupes électrogènes avant l'arrêt de la centrale électrique. Les usines de production pourraient alors achever leur cycle de production de façon nominale avant d'être stoppées évitant ainsi d'endommager les machines et les infrastructures.

	Critère	**Contrainte**
1	Efficacité, Confort, Coût	Never *aideInternationale*
2	Efficacité, Confort, Coût	-
3	Efficacité, Confort, Coût, Entreprise	-

Tableau 4.3 – Modèles de préférences du cas d'étude

La construction d'une fonction d'utilité via le logiciel MYRIAD est illustrée ci-dessous pour le premier modèle de préférences. Celle-ci débute par la création de l'arbre de préférences du modèle comme présenté sur la figure 4.15.

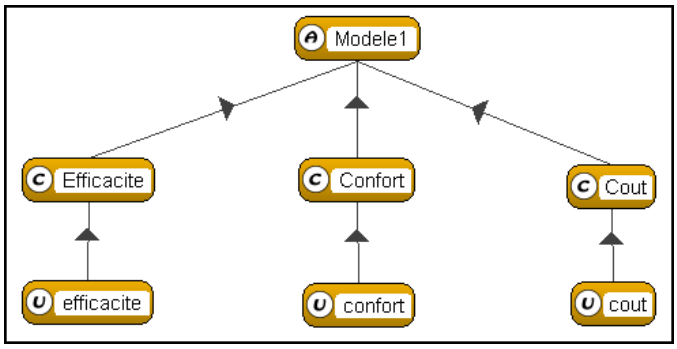


FIGURE 4.15 – Arbre de préférence du modèle 1 dans MYRIAD [95]

Il convient ensuite de déterminer la fonction d'utilité associée à chaque critère. Pour cela, le logiciel MYRIAD met en œuvre une procédure de questionnement analogue à celle décrite dans la section 2.4.1. La figure 4.16 présente les réponses des décideurs pour le critère de coût et la fonction d'utilité partielle correspondante construite par l'outil.

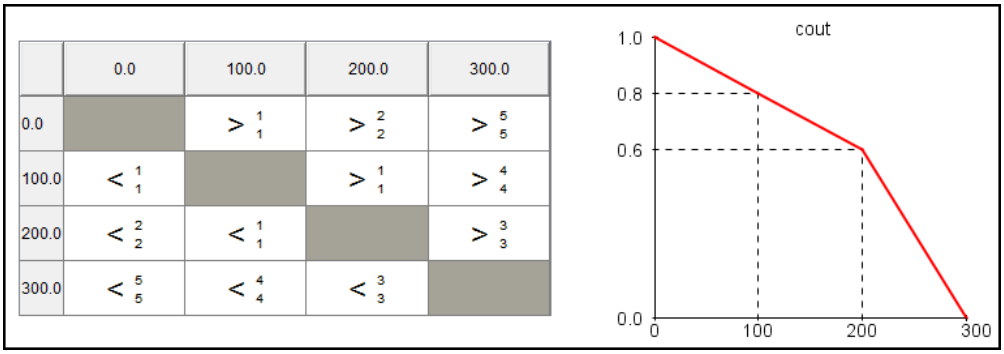


FIGURE 4.16 – Construction d'un critère dans MYRIAD [95]

Après que toutes les fonctions d'utilités partielles aient été déterminées, il faut construire la fonction d'agrégation du modèle de préférences comme mentionné dans la section 2.4.2. La figure 4.17 présente les réponses des décideurs à la procédure de questionnement proposée par MYRIAD. La fonction de capacité (dont l'interprétation est précisée dans la section 4.3.3) correspondant aux choix des décideurs est présentée dans le tableau 4.4.

	{}	Efficacite	Confort& Efficacite	Cout& Efficacite	Confort	Cout& Confort	Cout
{}		$<^{4}_{3}$	$<^{5}_{5}$	$<^{5}_{4}$	$<^{2}_{2}$	$<^{3}_{3}$	$<^{2}_{2}$
Efficacite	$>^{4}_{3}$		$<^{3}_{2}$	$<^{2}_{2}$	$>^{3}_{2}$	$>^{2}_{1}$	$>^{3}_{2}$
Confort& Efficacite	$>^{5}_{5}$	$>^{3}_{2}$		$>^{2}_{1}$	$>^{4}_{4}$	$>^{3}_{3}$	$>^{4}_{4}$
Cout& Efficacite	$>^{5}_{4}$	$>^{2}_{2}$	$<^{2}_{1}$		$>^{4}_{3}$	$>^{3}_{2}$	$>^{4}_{3}$
Confort	$>^{2}_{2}$	$<^{3}_{2}$	$<^{4}_{4}$	$<^{4}_{3}$		$<^{2}_{2}$	=
Cout& Confort	$>^{3}_{3}$	$<^{2}_{1}$	$<^{3}_{3}$	$<^{3}_{2}$	$>^{2}_{2}$		$>^{2}_{2}$
Cout	$>^{2}_{2}$	$<^{3}_{2}$	$<^{4}_{4}$	$<^{4}_{3}$	=	$<^{2}_{2}$	

FIGURE 4.17 – Construction d'une fonction d'agrégation dans MYRIAD [95]

$\mathfrak{P}(P)$	$\emptyset$	{1}	{2}	{3}	{12}	{13}	{23}	{123}
μ	0	0.5	0.2	0.2	0.8	0.7	0.4	1
m	0	0.5	0.2	0.2	0.1	0	0	0

TABLEAU 4.4 – Fonction de capacité correspondant aux choix des décideurs. Les critères 1, 2 et 3 sont les critères d'efficacité, de confort et de coût.

Une fois les modèles de préférences réalisés, ces derniers peuvent être importés dans le prototype comme illustré par la figure 4.18. Les partenaires peuvent alors préciser l'impact de leurs capacités sur les préférences numériques des modèles de préférences retenus pour adresser le problème.

Transformation de modèles

Les mécanismes de transformation de modèles n'ont été implémentés que partiellement dans le cadre de cette étude. En conséquence, la transformation des modèles opérationnels en problèmes PDDL a été réalisée manuellement pour ce cas d'étude.

Preference Model	Type	Status	Remove
Model 1	Optimization	✓	🗑
Model 2	Optimization	✓	🗑
Model 3	Optimization	✓	🗑

Partner	Capability	Efficacité	Confort	Coût (kEuros)
Pompiers - Pays-Bas	Eteindre incendie faible ampleur			
Pompiers - Pays-Bas	Eteindre incendie grande ampleur			
Pompiers - Pays-Bas	Réaliser évacuation	-	-	-
Pompiers - Pays-Bas	Réaliser évacuation par bateau	-	-	-
Pompiers - Pays-Bas	Réaliser évacuation par hélicoptère	-	-	-
Pompiers - Pays-Bas	Réaliser évacuation blessé	-	-	-
Pompiers - Pays-Bas	Renforcer digue	-	-	30
Police - Pays-Bas	Rechercher victime	-	-	-
Police - Pays-Bas	Diffuser consignes sécurité	-	1	-
Police - Pays-Bas	Réguler trafic	-	-	-

« 1 2 3 » 10 25 50 100

FIGURE 4.18 – Import des modèles de préférences dans le prototype

4.3.3 Résolution du problème de planification

Cette section présente les trois étapes de résolution des problèmes de planification à savoir : (i) l'appel au planificateur CHOPLAN, (ii) le post-traitement des solutions et (iii) la mise en œuvre de mécanismes d'aide à la décision pour supporter les décideurs lors de la sélection du plan à déployer.

Planification et post-traitement des solutions

Le planificateur CHOPLAN est utilisé pour résoudre les problèmes PDDL générés lors de la phase de modélisation. Pour ce cas d'étude, CHOPLAN a disposé d'une minute de temps de calcul et a été paramétré pour employer la stratégie de résolution SR_1 - h_2 (cf. section 3.2). Pour chacun des trois problèmes considérés, seule la meilleure solution identifiée par le planificateur a été retenue (voir plans 4.19, 4.20 et 4.22). Comme mentionné dans la section 4.2.1, les mécanismes de post-traitement des plans n'ont pas été implémentés dans le prototype. En conséquence, le post-traitement des solutions retournées par CHOPLAN a été réalisé manuellement pour

ce cas d'étude (voir figure 4.21 pour un exemple). Cette étape a pour objectif de paralléliser les actions du plans lorsque cela est possible. Il convient de préciser qu'il ne s'agit pas d'un réel ordonnancement du plan puisque le temps d'exécution des différentes actions du plan n'est pas connu.

Plan 1

1. *Diffuser_Consignes_Securite pb-police-equipe2*
2. *Installer_Lits_Centre_Accueil*
3. *Mettre_A_Disposition_Helicoptere*
4. *Determiner_Plan_Deroutement pb-police-equipe1*
5. *Mettre_En_Place_Axe_Circulation_Prioritaire hopital1 pb-police-equipe1*
6. *Realiser_Evacuation_Blesse_Hopital victime17 pb-pompier-equipe3*
7. *Realiser_Evacuation_Blesse_Hopital victime11 pb-pompier-equipe4*
8. *Realiser_Evacuation_Blesse_Hopital victime09 pb-pompier-equipe5*
9. *Realiser_Evacuation_Blesse_Hopital victime20 pb-pompier-equipe3*
10. *Realiser_Evacuation_Blesse_Hopital victime05 pb-pompier-equipe4*
11. *Realiser_Evacuation_Blesse_Hopital victime04 pb-pompier-equipe5*
12. *Mettre_En_Place_Axe_Circulation_Prioritaire centreacceuil1 pb-police-equipe2*
13. *Approvisionner_Nourriture_Centre_Acceuil*
14. *Installer_Dispensaire_Centre_Acceuil pb-croixrouge-equipe1*
15. *Ouvrir_Centre_Acceuil_Provisoire*
16. *Transferer_Patient*
17. *Mobiliser_Personnel_Garde*
18. *Renforcer_Digue pb-police-equipe1*
19. *Approvisionner_Pour_Etat_Alerte*
20. *Declarer_Hopital_En_Etat_Alterte*
21. *Reguler_Trafic pb-police-equipe2*
22. *Soigner_Blesse victime05*
23. *Soigner_Blesse victime09*
24. *Soigner_Blesse victime11*
25. *Soigner_Blesse victime17*
26. *Soigner_Blesse victime20*
27. *Rechercher_Victime victime14 pb-police-equipe1*
28. *Realiser_Evacuation_Par_Bateau victime07 pb-pompier-equipe3*
29. *Realiser_Evacuation_Par_Bateau victime12 pb-pompier-equipe3*
30. *Realiser_Evacuation_Par_Bateau victime13 pb-pompier-equipe3*
31. *Soigner_Blesse victime04*
32. *Rechercher_Victime victime15 pb-police-equipe1*
33. *Realiser_Evacuation victime19 pb-pompier-equipe4*
34. *Realiser_Evacuation victime18 pb-pompier-equipe5*
35. *Realiser_Evacuation_Par_Helicoptere victime03 pb-pompier-equipe4*
36. *Realiser_Evacuation victime10 pb-pompier-equipe5*
37. *Rechercher_Victime victime16 pb-police-equipe1*
38. *Realiser_Evacuation victime08 pb-pompier-equipe2*
39. *Realiser_Evacuation victime06 pb-pompier-equipe2*
40. *Realiser_Evacuation victime02 pb-pompier-equipe4*
41. *Realiser_Evacuation victime01 pb-pompier-equipe5*
42. *Alimenter_Groupe_Electrogene_Hopital*
43. *Arreter_Centrale_Electrique*

FIGURE 4.19 – Plan obtenu en optimisant le modèle de préférences 1

Plan 2

1. *Demander_Aide_Internationale*
2. *Diffuser_Consignes_Securite pb-police-equipe2*
3. *Installer_Lits_Centre_Acceuil*
4. *Mettre_A_Disposition_Helicoptere*
5. *Determiner_Plan_Deroutement pb-police-equipe1*
6. *Mettre_En_Place_Axe_Circulation_Prioritaire hopital1 pb-police-equipe1*
7. *Realiser_Evacuation_Blesse_Hopital victime17 pb-pompier-equipe3*
8. *Realiser_Evacuation_Blesse_Hopital victime11 pb-pompier-equipe4*
9. *Realiser_Evacuation_Blesse_Hopital victime09 pb-pompier-equipe5*
10. *Realiser_Evacuation_Blesse_Hopital victime20 pb-pompier-equipe3*
11. *Realiser_Evacuation_Blesse_Hopital victime05 pb-pompier-equipe4*
12. *Realiser_Evacuation_Blesse_Hopital victime04 pb-pompier-equipe5*
13. *Mettre_En_Place_Axe_Circulation_Prioritaire centreacceuil1 pb-police-equipe2*
14. *Approvisionner_Nourriture_Centre_Acceuil*
15. *Installer_Dispensaire_Centre_Acceuil pb-croixrouge-equipe1*
16. *Ouvrir_Centre_Acceuil_Provisoire*
17. *Transferer_Patient*
18. *Mobiliser_Personnel_Garde*
19. *Renforcer_Digue pb-police-equipe1*
20. *Approvisionner_Pour_Etat_Alerte*
21. *Declarer_Hopital_En_Etat_Alterte*
22. *Reguler_Trafic pb-police-equipe2*
23. *Soigner_Blesse victime05*
24. *Soigner_Blesse victime09*
25. *Soigner_Blesse victime11*
26. *Soigner_Blesse victime17*
27. *Soigner_Blesse victime20*
28. *Rechercher_Victime victime14 pb-police-equipe1*
29. *Realiser_Evacuation_Par_Bateau victime07 pb-pompier-equipe3*
30. *Realiser_Evacuation_Par_Bateau victime12 pb-pompier-equipe3*
31. *Realiser_Evacuation_Par_Bateau victime13 pb-pompier-equipe3*
32. *Soigner_Blesse victime04*
33. *Rechercher_Victime victime15 pb-police-equipe1*
34. *Realiser_Evacuation victime19 pb-pompier-equipe4*
35. *Realiser_Evacuation victime18 pb-pompier-equipe5*
36. *Realiser_Evacuation_Par_Helicoptere victime03 pb-pompier-equipe4*
37. *Realiser_Evacuation victime10 pb-pompier-equipe5*
38. *Rechercher_Victime victime16 pb-police-equipe1*
39. *Realiser_Evacuation victime08 pb-pompier-equipe2*
40. *Realiser_Evacuation victime06 pb-pompier-equipe2*
41. *Realiser_Evacuation victime02 pb-pompier-equipe4*
42. *Alimenter_Groupe_Electrogene_Hopital*
43. *Arreter_Centrale_Electrique*
44. *Mettre_A_Disposition_Nourriture*
45. *Distribuer_Nourriture_Zone_A_Risque al-croixrouge-equipe1*
46. *Realiser_Evacuation victime01 pb-pompier-equipe5*

FIGURE 4.20 – Plan obtenu en optimisant le modèle de préférences 2. Les actions en bleu mettent en exergue les différences avec le plan 1.

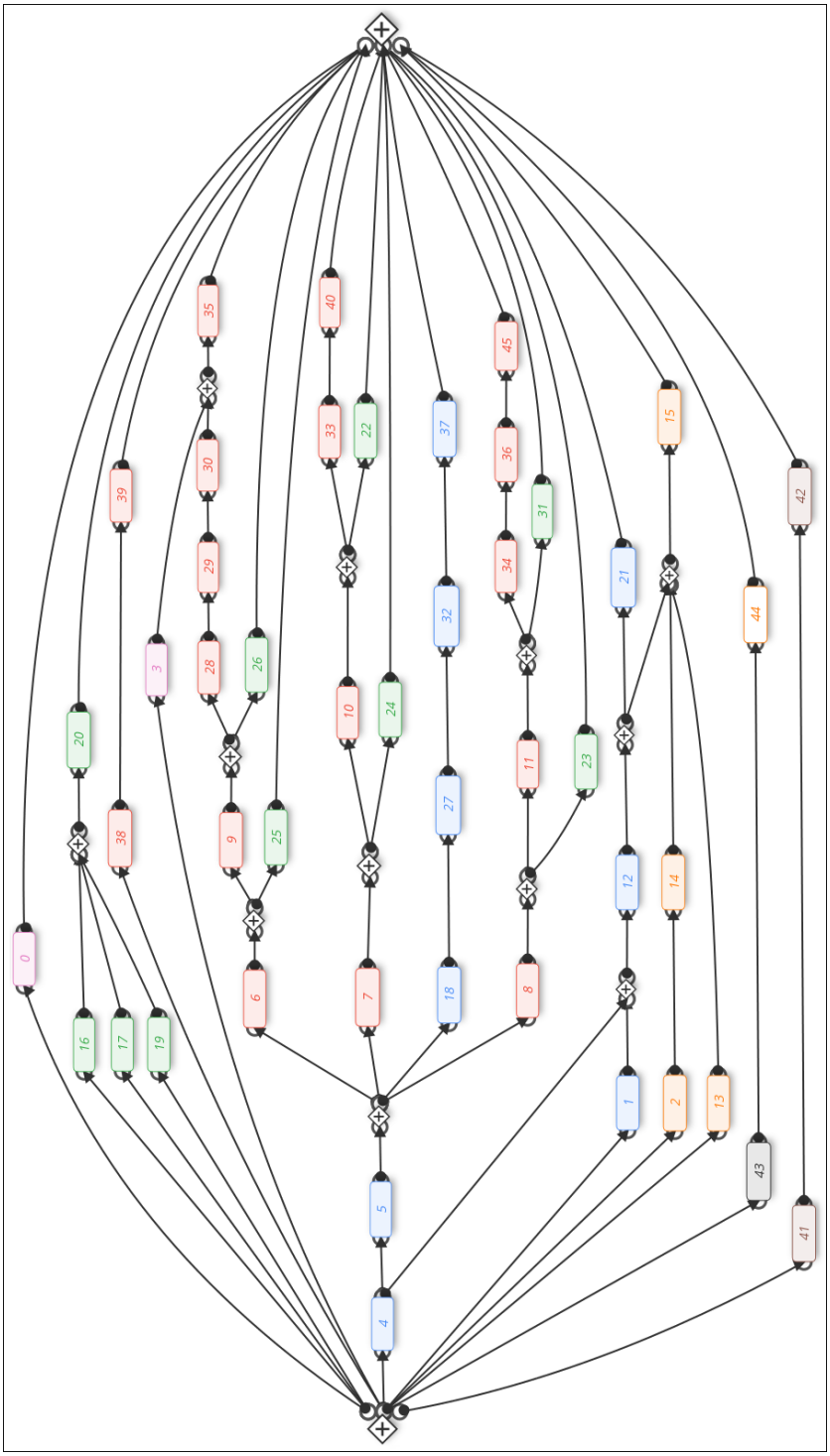


Figure 4.21 – Illustration des mécanismes de post-traitement sur le plan 2. Chaque couleur correspond aux actions d'un partenaire différent.

Plan 3

1. *Demander_Aide_Internationale*
2. *Diffuser_Consignes_Securite pb-police-equipe2*
3. *Installer_Lits_Centre_Acceuil*
4. *Mettre_A_Disposition_Helicoptere*
5. *Alimenter_Groupe_Electrogene_Entreprise*
6. *Determiner_Plan_Deroutement pb-police-equipe1*
7. *Mettre_En_Place_Axe_Circulation_Prioritaire hopital1 pb-police-equipe1*
8. *Realiser_Evacuation_Blesse_Hopital victime17 pb-pompier-equipe3*
9. *Realiser_Evacuation_Blesse_Hopital victime11 pb-pompier-equipe4*
10. *Realiser_Evacuation_Blesse_Hopital victime09 pb-pompier-equipe5*
11. *Realiser_Evacuation_Blesse_Hopital victime20 pb-pompier-equipe3*
12. *Realiser_Evacuation_Blesse_Hopital victime05 pb-pompier-equipe4*
13. *Realiser_Evacuation_Blesse_Hopital victime04 pb-pompier-equipe5*
14. *Mettre_En_Place_Axe_Circulation_Prioritaire centreacceuil1 pb-police-equipe2*
15. *Approvisionner_Nourriture_Centre_Acceuil*
16. *Installer_Dispensaire_Centre_Acceuil pb-croixrouge-equipe1*
17. *Ouvrir_Centre_Acceuil_Provisoire*
18. *Transferer_Patient*
19. *Mobiliser_Personnel_Garde*
20. *Renforcer_Digue pb-police-equipe1*
21. *Approvisionner_Pour_Etat_Alerte*
22. *Declarer_Hopital_En_Etat_Alterte*
23. *Reguler_Trafic pb-police-equipe2*
24. *Soigner_Blesse victime05*
25. *Soigner_Blesse victime09*
26. *Soigner_Blesse victime11*
27. *Soigner_Blesse victime17*
28. *Soigner_Blesse victime20*
29. *Rechercher_Victime victime14 pb-police-equipe1*
30. *Realiser_Evacuation_Par_Bateau victime07 pb-pompier-equipe3*
31. *Realiser_Evacuation_Par_Bateau victime12 pb-pompier-equipe3*
32. *Realiser_Evacuation_Par_Bateau victime13 pb-pompier-equipe3*
33. *Soigner_Blesse victime04*
34. *Rechercher_Victime victime15 pb-police-equipe1*
35. *Realiser_Evacuation victime19 pb-pompier-equipe4*
36. *Realiser_Evacuation victime18 pb-pompier-equipe5*
37. *Realiser_Evacuation_Par_Helicoptere victime03 pb-pompier-equipe4*
38. *Realiser_Evacuation victime10 pb-pompier-equipe5*
39. *Rechercher_Victime victime16 pb-police-equipe1*
40. *Realiser_Evacuation victime08 pb-pompier-equipe2*
41. *Realiser_Evacuation victime06 pb-pompier-equipe2*
42. *Realiser_Evacuation victime02 pb-pompier-equipe4*
43. *Alimenter_Groupe_Electrogene_Hopital*
44. *Arreter_Centrale_Electrique*
45. *Mettre_A_Disposition_Nourriture*
46. *Distribuer_Nourriture_Zone_A_Risque al-croixrouge-equipe1*
47. *Realiser_Evacuation victime01 pb-pompier-equipe5*

FIGURE 4.22 – Plan obtenu en optimisant le modèle de préférences 3. Les actions en bleu mettent en exergue les différences avec le plan 2.

Aide à la décision

Afin d'aider les décideurs à choisir le plan à exécuter, le prototype présente les scores obtenus par les différents plans solutions par rapport à chacun des modèles de préférences considérés (cf. figure 4.23).

Summary	Model 1	Model 2	Model 3	Min	Max	Average
Alternative 1	**58 %**	58 %	58 %	58 %	58 %	58 %
Alternative 2	0 %	**81 %**	81 %	0 %	81 %	54 %
Alternative 3	0 %	78 %	**86 %**	0 %	86 %	55 %

Figure 4.23 – Résumé des scores des plans solutions dans le cas d'étude

Les modèles de préférences 1 et 2 ne diffèrent que par la contrainte relative à l'utilisation de l'aide internationale. Seule l'alternative 1 (qui a été obtenue en cherchant à optimiser le modèle 1) respecte cette contrainte ce qui explique la note nulle des plans 2 et 3 par rapport au modèle de préférence 1. Les scores des alternatives 1 et 2 par rapport au modèle 2 peuvent être interprétés à l'aide des diagrammes en secteurs de la figure 4.24. Conformément aux choix des décideurs lors de la procédure de questionnement (cf. figure 4.17), l'efficacité de la réponse est le critère prépondérant du modèle puisqu'il est responsable de 50% du score. De même, les critères de confort et de coût impactent tous deux 20% de la note. Les 10% du score restant sont dus à une complémentarité entre les critères d'efficacité et de confort. Ainsi, cette section de la note favorise les plans qui sont satisfaisants sur ces deux critères conjointement. C'est par exemple le cas de l'alternative 2 dont les scores d'efficacité et de confort sont relativement similaires (cf. diagrammes à barres de la figure 4.24) mais pas de l'alternative 1 dont le score du critère d'efficacité est largement supérieur à celui du critère de confort.

Les figures 4.19 et 4.20 permettent de comprendre les différences de scores entre les alternatives 1 et 2. En effet, le plan 2 se distingue du premier plan par son recours à l'aide internationale offerte par l'Allemagne et la Belgique. En conséquence, le nombre d'équipes de secours mobilisées dans le plan 2 est supérieur ce qui impacte positivement son critère d'efficacité. Les effectifs de la Croix Rouge sont ainsi renforcés ce qui permet à cet acteur d'effectuer une mission supplémentaire dans le plan 2 (distribution de nourriture aux habitants de la zone qui va être isolée) par

rapport à l'unique mission qu'elle assume dans le plan 1 (prise en charge de blessés dans le dispensaire du centre d'accueil). En conséquence, l'alternative 2 obtient un bon score par rapport au critère de confort. En revanche, le score de coût du plan 2 est plus faible que celui du plan 1 à cause des frais d'approvisionnement occasionnés par cette distribution de nourriture.

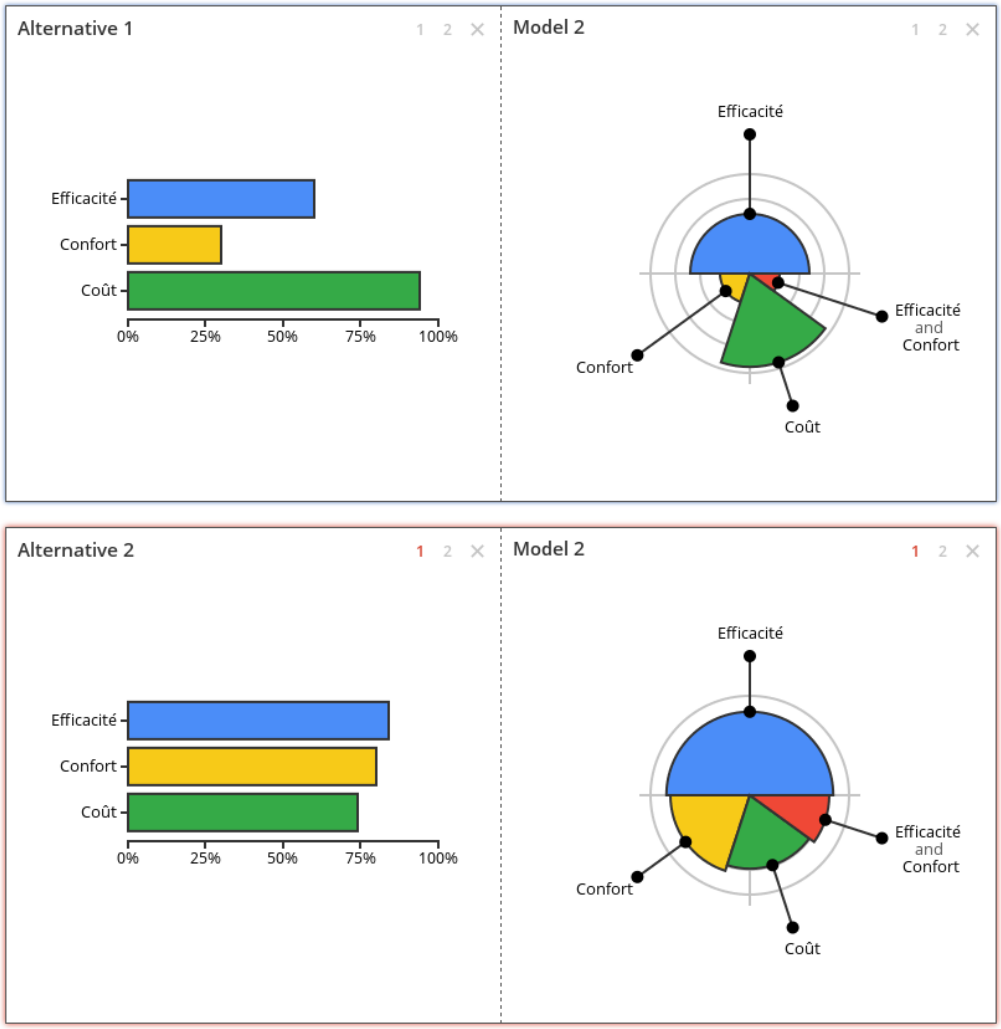


FIGURE 4.24 – Comparaison des plans 1 et 2 selon le modèle de préférences 2

L'analyse des résultats peut se poursuivre à l'aide de la figure 4.25 qui compare les alternatives 2 et 3 par rapport au modèle de préférences 3. Ce dernier présente des similarités avec les modèles de préférences 1 et 2 puisque le critère d'efficacité représente 50% de la note, le critère de coût représente 20% de la note et la

complémentarité entre les critères d'efficacité et de confort représente également 10% du score. Néanmoins, les 20% restants de la note sont définis par une interaction négative entre les critères de coût et d'entreprise. En effet, dans ce modèle de préférences, ces deux critères sont complètement substituables l'un à l'autre. Ainsi, un bon score sur le critère d'entreprise peut compenser un mauvais coût de la solution et vice-versa.

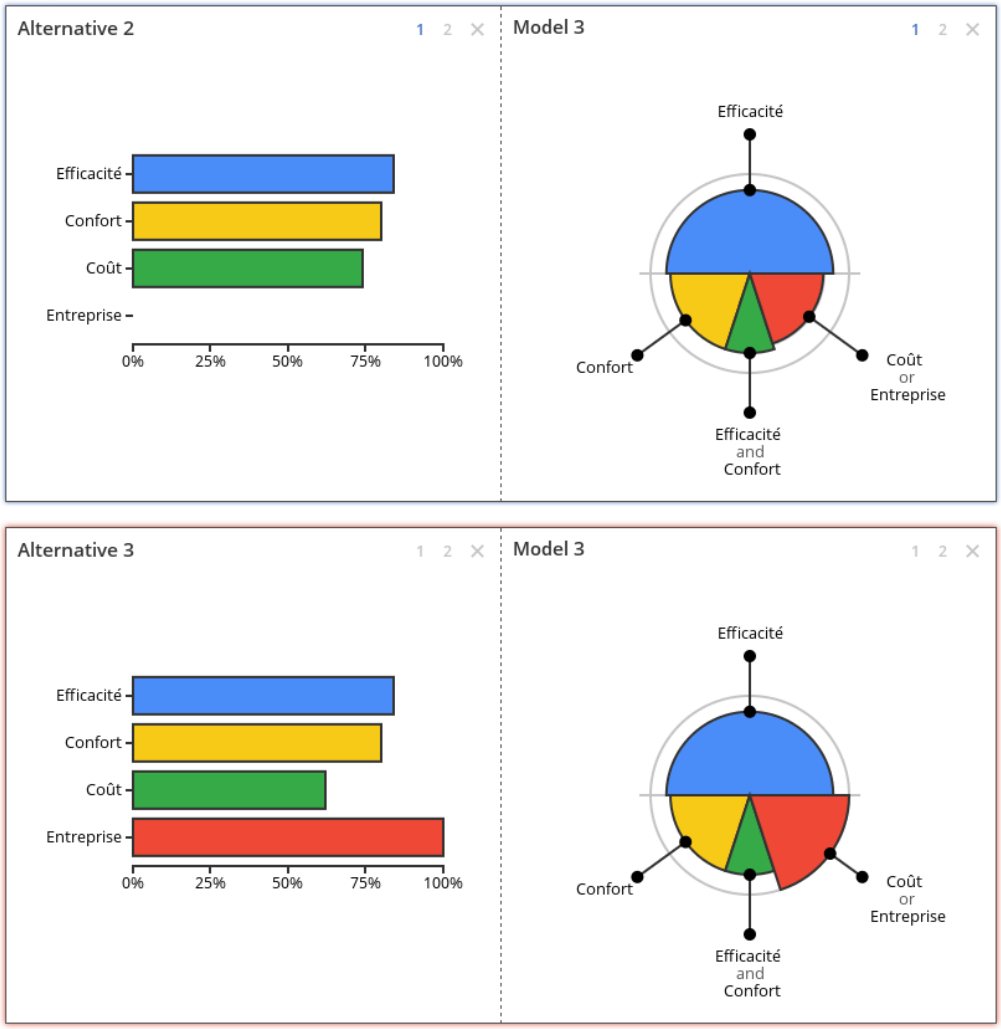


FIGURE 4.25 – Comparaison des plans 2 et 3 selon le modèle de préférences 3

L'alternative 3 (qui est obtenue en optimisant le troisième modèle de préférences) se distingue de l'alternative 2 par le fait qu'elle respecte la préférence relative à l'alimentation électrique des usines de l'entreprise (cf. figure 4.22). Par conséquent,

le score du plan 3 selon le critère entreprise est maximal. En revanche, son critère de coût est moins bon que celui du plan 2 à cause des frais liés à l'alimentation électrique de l'entreprise. Ces deux critères étant substituables l'un à l'autre, l'alternative 3 surpasse l'alternative 2 au sens du troisième modèle de préférences. La qualité du plan 2 reste néanmoins supérieure à celle du plan 3 en ce qui concerne le modèle de préférences 2.

Dans ce cas d'étude, les décideurs choisissent finalement de déployer le troisième plan. En effet, il leur semble difficile de se passer de la présence des partenaires internationaux au vue de l'impact de ces derniers sur la qualité de la réponse. Par ailleurs, le surcout associé au troisième plan leur semble acceptable au vue des avantages de ce dernier par rapport au deuxième plan.

Conclusion

Ce chapitre a débuté par une introduction du domaine de la gestion de crise. Il a notamment présenté les différentes problèmes d'interopérabilité auxquels peuvent être confrontés des organisations qui collaborent dans le but de résoudre une situation de crise. Un système d'aide à la décision qui adresse ces difficultés via la construction de plans d'action collaboratifs a été présenté. Il permet à un aréopage d'entités organisationnelles de représenter le problème qu'elles souhaitent résoudre collectivement puis leur propose un ou plusieurs plans collaboratifs dont l'exécution permettrait d'atteindre leurs objectifs. Pour cela, ce dernier met en œuvre des mécanismes de représentation de la connaissance et de transformation de modèles ainsi que les mécanismes de planification présentés dans le chapitre 3. Par ailleurs, un prototype de ce système a été développé dans le cadre de cette étude. Son utilisation a été illustrée à l'aide d'un exemple dans lequel plusieurs partenaires sont amenés à travailler collectivement pour faire face à une inondation et à ses conséquences. Ce prototype met en exergue d'une part l'intérêt et d'autre part la faisabilité conceptuelle du système d'aide à la décision proposé. Il reste néanmoins perfectible notamment en ce qui concerne la transformation des modèles opérationnels des décideurs en problèmes de planification PDDL qui n'a été que partiellement implémentée.

Conclusion générale

Ces travaux se sont intéressés à la problématique de la planification avec préférences basée sur un modèle MAUT couplé à une intégrale de Choquet. Les résultats obtenus ont été appliqués au domaine de la gestion de crise via le prototypage d'un système capable d'accompagner des décideurs lors de la création de plans d'action collaboratifs de réponse à une situation de crise. Cette conclusion générale rappelle les contributions de cette étude. De plus, quelques perspectives scientifiques relatives à ces contributions sont également proposées.

Contribution à l'amélioration du pouvoir expressif du PDDL

Une exigence PDDL nommée `maut-preferences` a été spécifiée dans le cadre de cette étude. Elle généralise la notion de préférence utilisée en PDDL3 en encodant ces dernières au sein de critères préférentiels issus d'un modèle MAUT (Théorie de l'Utilité Multi-Attribut) couplé à une intégrale de Choquet. Cette exigence enrichit le pouvoir expressif du langage PDDL3 puisqu'elle permet notamment d'employer des préférences floues, d'utiliser facilement un nombre quelconques de préférences numériques, d'agréger des préférences élémentaires entre elles ou encore de considérer d'éventuelles interactions entre préférences. En outre, elle offre naturellement la possibilité de mettre en œuvre simplement des techniques d'aide à la décision permettant de construire des modèles de préférences MAUT uniquement à partir d'informations préférentielles opérationnelles. Ceci est particulièrement intéressant pour certains cas applicatifs.

Les perspectives envisageables quant à l'amélioration du pouvoir expressif du PDDL consisteraient à étendre l'exigence `maut-preferences` pour prendre en compte les préférences temporelles qui n'ont pas été considérées dans cette étude. Il serait ainsi possible d'utiliser les éléments PDDL3 de type (`within` t ϕ), (`hold-after` t ϕ)

et (`hold-during` t_1 t_2 ϕ) [64] qui permettent de formaliser des préférences relatives au temps dans les plans solutions notamment le fait qu'un prédicat ϕ doit être vrai avant ou après le temps t ou encore de t_1 à t_2.

Contribution à la problématique de la planification avec préférences

Un algorithme de planification avec préférences a été mis en œuvre dans le cadre de ces travaux. Ce dernier s'appuie sur des heuristiques qui évaluent le potentiel des nœuds à l'aide de deux estimations respectivement relatives aux objectifs à atteindre et aux préférences du problème. L'originalité de ces heuristiques consiste à agréger ces deux estimations à l'aide d'une intégrale de Choquet. Cet algorithme a été utilisé pour implémenter un planificateur nommé ChoPlan dont les performances ont été comparées à celles des planificateurs de l'art sur un ensemble de problèmes de référence. Les tests réalisés suggèrent que l'approche retenue pour résoudre les problèmes de planification avec préférences est efficace.

Trois perspectives sont proposées quant à l'amélioration de ces résultats. La première consisterait à étendre le support du langage PDDL dans ChoPlan. En effet, en gérant l'exigence PDDL `existential-preconditions`, ce dernier serait en mesure de résoudre davantage de problèmes de référence de la communauté de la planification. Par ailleurs, une règle de coupe basée sur la notion de Choquet-dominance [60] pourrait être implémentée dans ChoPlan. La Choquet-dominance est particulièrement intéressante puisqu'elle permet d'éliminer plus de nœuds que la Pareto-dominance lors de la recherche. Enfin, il serait intéressant de construire une heuristique h_4 basée sur les estimations Δ_1 et Λ_4 avec Λ_4 modélisant une substituabilité parfaite entre les estimations Λ_1 et Λ_2. Cette heuristique pourrait possiblement combiner les avantages des heuristiques h_1 et h_2 de ce manuscrit.

Contribution au domaine de la gestion de crise

Dans le cadre de ces travaux, un système d'aide à la décision capable de supporter les décideurs de la gestion de crise lors de l'élaboration de plans d'action collaboratifs a été spécifié. Ce dernier s'intègre au sein de la démarche MISE qui propose ensuite de déployer le plan construit sur un système d'information de médiation puis de

monitorer son exécution. Un prototype de ce système a été réalisé et utilisé pour résoudre un cas d'étude issu du projet européen SECTOR. Ce dernier constitue une preuve de concept du fonctionnement du système proposé.

Le prototype réalisé n'implémente que partiellement le système d'aide à la décision décrit, la transformation des modèles opérationnels en problèmes PDDL et le post-traitement des plans n'étant pas entièrement automatisés. Le développement de ces éléments constitue donc une suite logique à cette étude. De plus, des travaux sur la capitalisation de connaissances opérationnelles de type retours d'expériences ou bonnes pratiques pourraient être conduits. Ces connaissances pourraient alors être employées lors de l'étape d'inférence pour faciliter la modélisation du problème aux décideurs. En outre, ces dernières pourraient également être utilisées pour implémenter dans ChoPlan des mécanismes de résolution spécifiques à la gestion de crise (nouveau type de préférences élémentaires, règles de sélection et règles de coupes basées sur la doctrine et les retours d'expériences...). Une troisième perspective concernant le domaine de la gestion de crise consisterait à rapprocher les mécanismes d'agilité du projet MISE avec la notion de préférence qui est au cœur de ces travaux. Ces mécanismes permettent de re-exécuter le processus de planification lorsque des divergences sont constatées entre le plan théorique et sa mise en œuvre dans le monde réel. Une problématique opérationnelle apparait alors : comment assurer la continuité entre les actions actuellement en cours d'exécution et les actions du plan nouvellement défini ? Une solution pourrait consister à modéliser ces nouvelles contraintes à respecter comme des préférences des décideurs.

Pour finir, il convient de rappeler que les résultats de ces travaux ont une portée générique et peuvent être utilisés dans de nombreux domaines applicatifs autres que celui de la gestion de crise. A titre d'exemple, ils ont également été employés pour mener une étude préliminaire sur une problématique de planification d'activités de développement et de déploiement des éléments constitutifs d'un système de système multi-industriels. Ce problème intéresse fortement la société Thales Communications and Security et peut bénéficier des résultats de cette étude quant au pouvoir expressif avec lequel les décideurs peuvent formaliser leurs préférences.

Bibliographie

[1] AFUL, Groupe de travail sur l'interopérabilité : Définition de l'interopérabilité. `http://definition-interoperabilite.info`.

[2] N. Altay et W. G. Green : OR/MS research in disaster operations management. *European Journal of Operational Research*, 175(1):475–493, 2006.

[3] F. Bacchus et F. Kabanza : Using temporal logic to express search control knowledge for planning. *Artificial Intelligence*, 116(1):123–191, 2000.

[4] J. W. Backus, F. L. Bauer, J. Green, C. Katz, J. McCarthy, A. J. Perlis, H. Rutishauser, K. Samelson, B. Vauquois, J. H. Wegstein, A. van Wijngaarden et M. Woodger : Revised Report on the Algorithm Language ALGOL 60. *Communications of the ACM*, 6(1):1–17, 1963.

[5] J. Baier et S. A. McIlraith : On Domain-Independent Heuristics for Planning with Qualitative Preferences. *Dans AAAI Spring Symposium : Logical Formalizations of Commonsens Reasoning*, p. 7–12, 2007.

[6] J. A. Baier, F. Bacchus et S. A. McIlraith : A Heuristic Search Approach to Planning with Temporally Extended Preferences. *Dans Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, p. 1808–1815, 2007.

[7] J. A. Baier et S. A. McIlraith : Planning with Preferences. *AI Magazine*, 29(4):25, 2008.

[8] C. A. Bana e Costa, J.-M. De Corte et J.-C. Vansnick : *On the mathematical foundation of MACBETH*. Springer, 2005.

[9] C. A. Bana e Costa et J.-C. Vansnick : A Theorical Framework for Measuring Attractiveness by a Categorical Based Evaluation Technique (MACBETH). *Dans Multicriteria Analysis*, p. 15–24. Springer, 1997.

[10] C. A. Bana e Costa et J.-C. Vansnick : *The MACBETH approach : Basic Ideas, Software, and an Application*. Springer, 1999.

[11] A. Barret, D. Christianson, M. Friedman, K. Golden, C. Kwok, J. Penberthy, Y. Sun et D. Weld : UCPOP user's manual. Rap. tech., 93-09-06d, University of Washington, 1995.

[12] A.-M. Barthe-Delanoë, S. Truptil, F. Bénaben et H. Pingaud : Event-driven agility of interoperability during the Run-time of collaborative processes. *Decision Support Systems*, 59:171–179, 2014.

[13] F. Bénaben : *Conception de Système d'Information de Médiation pour la prise en charge de l'Interopérabilité dans les Collaborations d'Organisations*. Habilitation à diriger des recherches, Institut National Polytechnique de Toulouse, 2012.

[14] F. Bénaben, N. Boissel-Dallier, H. Pingaud et J.-P. Lorré : Semantic issues in model-driven management of information system interoperability. *International Journal of Computer Integrated Manufacturing*, 26(11):1042–1053, 2013.

[15] F. Bénaben, M. Lauras, S. Truptil et J. Lamothe : MISE 3.0 : An Agile Support for Collaborative Situation. *Dans Collaborative Networks in the Internet of Services*, vol. 380. Springer Berlin Heidelberg, 2012.

[16] F. Bénaben, M. Lauras, S. Truptil et N. Salatgé : A Metamodel for Knowledge Management in Crisis Management. *Dans 49th Hawai International Conference on System Sciences (HICSS)*, p. 126–135, 2016.

[17] F. Bénaben, W. Mu, N. Boissel-Dallier, A.-M. Barthe-Delanoë, S. Zribi et H. Pingaud : Supporting interoperability of collaborative networks through engineering of a service-based Mediation Information System (MISE 2.0). *Enterprise Information Systems*, 9(5-6):556–582, 2015.

[18] F. Bénaben, W. Mu, S. Truptil, H. Pingaud et J.-P. Lorré : Information systems design for emerging ecosystems. *Dans 4th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, 2010.

[19] C. L. Bennett, D. Larson, J. L. Weiland, N. Jarosik, G. Hinshaw, N. Odegard, K. M. Smith, R. S. Hill, B. Gold, M. Halpern, E. Komatsu, M. R. Nolta, L. Page, D. N. Spergel, E. Wollack, J. Dunkley,

A. Kogut, M. Limon, S. S. Meyer, G. S. Tucker et E. L. Wright : Nine-year Wilkinson Microwave Anisotropy Probe (WMAP) Observations : Final Maps and Results. *Astrophysical Journal Supplement Series*, 208, 2013.

[20] J. Benton, S. Kambhampati et M. B. Do : YochanPS : PDDL3 Simple Preferences and Partial Satisfaction Planning. *Paper presented in the Fifth International Planning Competition*, 2006.

[21] J. Benton, M. Van Den Briel et S. Kambhampati : A Hybrid Linear Programming and Relaxed Plan Heuristic for Partial Satisfaction Planning Problems. *Dans Proceedings of the 17th International Conference on Automated Planning and Scheduling (ICAPS)*, p. 34–41, 2007.

[22] L. Bidoux, F. Bénaben et J.-P. Pignon : A Metamodel for Collaboration Formalization. *Dans Collaborative Systems for Smart Networked Environments*, p. 375–383, 2014.

[23] M. Bienvenu, C. Fritz et S. A. McIlraith : Planning with Qualitative Temporal Preferences. *KR*, 6:134–144, 2006.

[24] P. Blackburn, J. van Benthem et F. Wolter : *Handbook of Modal Logic*, vol. 3. Elsevier, 2006.

[25] A. Blum et M. L. Furst : Fast planning through planning graph analysis. *Dans Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, p. 1636–1642, 1995.

[26] A. L. Blum et M. L. Furst : Fast planning through planning graph analysis. *Artificial intelligence*, 90(1):281–300, 1997.

[27] B. Bonet et H. Geffner : Planning as heuristic search : New results. *Dans Recent Advances in AI Planning*, p. 360–372. Springer, 1999.

[28] B. Bonet et H. Geffner : Planning as heuristic search. *Artificial Intelligence*, 129(1):5–33, 2001.

[29] C. Boutilier, R. Brafman, C. Domshlak, H. H. Hoos et D. Poole : CP-nets : A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. *Journal of Artificial Intelligence Research*, 21:135–191, 2004.

[30] C. Boutilier, T. Dean et S. Hanks : Decision-Theoric Planning : Structural Assumptions and Computational Leverage. *Journal of Artificial Intelligence Research*, 11(1):94, 1999.

[31] R. Brafman, C. Domshlak et S. E. Shimony : On Graphical Modeling of Preference and Importance. *Journal of Artificial Intelligence Research*, p. 389–424, 2006.

[32] G. Brewka, I. Niemela et M. Truszczynski : Preferences and Nonmonotonic Reasoning. *AI Magazine*, 29(4):69, 2009.

[33] J. R. Büchi : On a decision method in restricted second order arithmetic. *Dans Proc. Internat. Congr. Logic, Method. and Philos. Sci*, p. 1–12, 1960.

[34] CCA (Club de la Continuité d'Activité) : Lexique structuré de la continuité d'activité. Business continuity structured glossary. *Livre blanc du Club de la Continuité d'Activité*, 2014.

[35] K. Chiba, Y. Makino et T. Takatoya : Design-Informatics Approach Applicable to Real-World Problem. *Dans Computational Intelligence in Multicriteria Decision-Making*, p. 167–174, 2011.

[36] G. Choquet : Theory of capacities. *Annales de l'Institut Fourier*, 5:131–295, 1953.

[37] I. Chriswell et W. Hodges : *Mathematical Logic*, vol. 3. Oxford University Press, 2007.

[38] M. Cinque, C. Esposito, M. Fiorentino, J. Mauthner, Ł. Szklarski, F. Wilson, Y. Semet et J.-P. Pignon : SECTOR : Secure Common Information Space for the Interoperability of First Responders. *Procedia Computer Science*, 64:750–757, 2015.

[39] A. J. Coles et A. Coles : LPRPG-P : Relaxed Plan Heuristics for Planning with Preferences. *Dans Proceedings of the 21st International Conference on Automated Planning and Scheduling (ICAPS)*, 2011.

[40] T. Dean et M. Wellman : *Planning and Control*. Morgan Kaufmann Publishers, 1991.

[41] R. Dechter, I. Meiri et J. Pearl : Constraint satisfaction. *Dans MIT Encyclopedia of the Cognitive Sciences (MITECS)*, 1998.

[42] L. Derek et M. Fox : The 3rd International Planning Competition : Results and Analysis. *Journal of Artificial Intelligence Research*, 20:1–59, 2003.

[43] M. Do, M. Helmert et I. Refanidis : Deterministic Track of the Sixth International Planning Competition. `http:\icaps-conference.org/ipc2008/deterministic/HomePage.html`.

[44] C. Domshlak, E. Hüllermeier, S. Kaci et H. Prade : Preferences in AI : An overview. *Artificial Intelligence*, 175(7):1037–1052, 2011.

[45] J. S. Dyer : MAUT - Multiattribute Utility Theory. *Dans Multiple Criteria Decision Analysis : State of the art surveys*, p. 265–292. Springer, 2005.

[46] A. S. Eddington : The Constants of Nature. *The World of Mathematics*, 2, 1956.

[47] S. Edelkamp : On the Compilation of Plan Constraints and Preferences. *Dans Proceedings of the 16th International Conference on Automated Planning and Scheduling (ICAPS)*, p. 374–377, 2006.

[48] S. Edelkamp : Optimal symbolic PDDL3 planning with MIPS-BDD. *5th International Planning Competition Booklet*, 2006.

[49] S. Edelkamp et J. Hoffman : PDDL2.2 : The Language for the Classical Part of the 4th International Planning Competition. *4th International Planning Competition (IPC'04)*, 2004.

[50] S. Edelkamp, S. Jabbar et M. Naizih : Large-scale optimal PDDL3 planning with MIPS-XXL. *5th International Planning Competition Booklet*, p. 28–30, 2006.

[51] S. Edelkamp et P. Kissmann : GAMER : Bridging Planning and General Game Playing with Symbolic Search. *6th International Planning Competition Booklet*, 2008.

[52] M. Ehrgott : *Multicriteria optimization*. Springer Science & Business Media, 2006.

[53] D. D. Eisenhower : Remarks at the National Defense Executive Reserve Conference, 14 Novembre 1957. *The American Presidency Project*. `http://www.presidency.ucsb.edu/ws/?pid=10951`.

[54] T. Erl : *Service-oriented architecture (SOA) : concepts, technology, and design*. Prentice Hall, 2005.

[55] S. Even : *Graph algorithms*. Cambridge University Press, 2011.

[56] R. Feldman, G. Brewka et S. Wenzel : Planning with Prioritized Goals. *Dans KR*, p. 513–514, 2006.

[57] J. Figueira, S. Greco et M. Ehrgott : *Multiple Criteria Decision Analysis : State of the art surveys*, vol. 78. Springer Science & Business Media, 2005.

[58] J. Figueira, S. Greco, B. Roy et R. Slowinski : ELECTRE Methods : Main Features and Recent Developments. *Dans Handbook of Multicriteria Analysis*, p. 51–89. Springer, 2010.

[59] J. Figueira, V. Mousseau et B. Roy : ELECTRE methods. *Dans Multiple criteria decision analysis : State of the art surveys*, p. 133–153. Springer, 2005.

[60] H. Fouchal : *Optimisation de l'intégrale de Choquet pour le calcul de plus courts chemins multi-objectifs préférés*. Thèse de doctorat, Université de Nantes, 2011.

[61] M. Fox et D. Long : PDDL2.1 : An Extension to PDDL for Expressing Temporal Planning Domains. *Journal of Artificial Intelligence Research*, 20:61–124, 2003.

[62] J. Garson : Modal Logic. *Dans The Stanford Encyclopedia of Philosophy*. `http://plato.stanford.edu/archives/sum2014/entries/logic-modal/`, Summer 2014.

[63] A. Gerevini et D. Long : BNF Description of PDDL3.0. *Unpublished manuscript from the IPC-5 website*, 2005.

[64] A. Gerevini et D. Long : Preferences and Soft Constraints in PDDL3. *Dans ICAPS workshop on planning with preferences and soft constraints*, p. 46–53, 2006.

[65] A. E. Gerevini, P. Haslum, D. Long, A. Saetti et Y. Dimopoulos : Deterministic planning in the fifth international planning competition : PDDL3 and experimental evaluation of the planners. *Artificial Intelligence*, 173(5):619–668, 2009.

[66] M. Ghallab, D. Nau et P. Traverso : *Automated Planning : Theory and Practice*. Morgan Kaufmann Publishers, 2004.

[67] J. Goldsmith et U. Junker : Preference Handling for Artificial Intelligence. *AI Magazine*, 29(4):9, 2009.

[68] M. Grabisch : The application of fuzzy integrals in Multicriteria Decision Making. *European Journal of Operational Research*, 89(3):445–456, 1996.

[69] M. Grabisch : Alternative representations of discrete fuzzy measures for decision making. *International Journal of Uncertainty, Fuziness and Knowledge-Based Systems*, 5(05):587–607, 1997.

[70] M. Grabisch : K-order additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems*, 92(2):167–189, 1997.

[71] M. Grabisch : Une approche constructive de la décision multicritère. *Traitement du signal*, 22(4):321–337, 2005.

[72] M. Grabisch, S. Greco et M. Pirlot : Bipolar and bivariate models in multicriteria decision analysis : Descriptive and constructive approaches. *International Journal of Intelligent Systems*, 23(9):930–969, 2008.

[73] M. Grabisch et C. 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.

[74] M. Grabisch et M. Roubens : Application of the Choquet Integral in Multicriteria Decision Making. *Fuzzy Measures and Integrals - Theory and Applications*, p. 348–374, 2000.

[75] T. R. Gruber : Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5): 907–928, 1995.

[76] N. Guedj : *Pour des casques rouges à l'ONU*. Le cherche midi. ISBN : 978-2-7491-1428-6, 2009.

[77] D. Guha-Sapir, P. Hoyois et R. Below : Annual Disaster Statistical Review 2014, the number and trends, 2014.

[78] P. Haslum : Additive and Reversed Relaxed Reachability Heuristics Revisited. *6th International Planning Competition Booklet*, 2008.

[79] J. Hoffmann : A heuristic for domain independent planning and its use in an enforced hill-climbing algorithm. *Dans Foundations of Intelligent Systems*, p. 216–227. Springer, 2000.

[80] J. Hoffmann : FF : The fast-forward planning system. *AI magazine*, 22(3):57, 2001.

[81] J. Hoffmann et B. Nebel : The FF planning system : Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, p. 253–302, 2001.

[82] R. Howey, D. Long et M. Fox : VAL : Automatic plan validation, continuous effects and mixed initiative planning using PDDL. *Dans Tools with Artificial Intelligence*, 2004.

[83] C.-W. Hsu, B. W. Wah, R. Huang et Y. Chen : Constraint Partitioning for Solving Planning Problems with Trajectory Constraints and Goal Preferences. *Dans Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, p. 1924–1929, 2007.

[84] A. Imoussaten, J. Montmain et G. Mauris : A Multicriteria Decision Support System using a Possibility Representation for Managing Inconsistent Assessments of Experts Involved in Emergency Situations. *International Journal of Intelligent Systems*, 29(1):50–83, 2014.

[85] E. Jacquet-Lagreze et J. 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.

[86] H. Kautz et B. Selman : Pushing the envelope : Planning, propositional logic, and stochastic search. *Dans Proceedings of the National Conference on Artificial Intelligence*, p. 1194–1201, 1996.

[87] H. A. Kautz et B. Selman : Planning as satisfiability. *Dans Proceedings of the European Conference on Artificial Intelligence (ECAI)*, vol. 92, p. 359–363, 1992.

[88] R. L. Keeney et H. Raiffa : *Decisions with Multiple Objectives : Preferences and Value Tradeoffs*. Cambridge University Press, 1993.

[89] A. G. Kleppe, J. B. Warmer et W. Bast : *MDA explained : the model driven architecture : practice and promise*. Addison-Wesley Professional, 2003.

[90] J. Koehler et J. Hoffmann : On the Instantiation of ADL Operators Involving Arbitrary First-Order Formulas. *Dans PuK*, 2000.

[91] D. L. Kovacs : BNF Definition of PDDL3.1. Unpublished manuscript from the IPC-2011 website, 2011.

[92] V. Kumar : Algorithms for constraint-satisfaction problems : A survey. *AI magazine*, 13(1):32, 1992.

[93] C. Labreuche et M. Grabisch : The Choquet integral for the aggregation of interval scales in multicriteria decision making. *Fuzzy Sets and Systems*, 137(1):11–26, 2003.

[94] C. Labreuche et M. Grabisch : Fuzzy Measures and Integrals in MCDA. *Dans Multiple Criteria Decision Analysis : State of the art surveys*, p. 563–608. Springer, 2005.

[95] C. Labreuche et F. Lehuédé : MYRIAD : a tool suite for MCDA. *Dans EUSFLAFT*, vol. 5, p. 204–209, 2005.

[96] C. Y. Lee : An algorithm for path connections and its applications. *IRE Transactions on Electronic Computers*, (3):346–365, 1961.

[97] F. Lehuédé, M. Grabisch, C. Labreuche et P. Savéant : Integration and Propagation of a Multi-Criteria Decision Making Model in Constraint Programming. *Journal of Heuristics*, 12(4-5):329–346, 2006.

[98] F. Lehuédé, M. Grabisch, C. Labreuche et P. Savéant : MCS-A new algorithm for multicriteria optimisation in constraint programming. *Annals of Operations Research*, 147(1):143–174, 2006.

[99] V. Lifschitz : On the semantics of STRIPS. *Dans Reasoning about Actions and Plans : Proceedings of the 1986 Workshop*, p. 1–9, 1987.

[100] L. Lovasz : Submodular functions and convexity. *Dans Mathematical Programming. The State of The Art*, p. 235–257. Springer, 1983.

[101] G. Macé-Ramète, F. Bénaben, M. Lauras et J. Lamothe : Suporting Decision for Road Crisis Management through an Agile and Collaborative Information System. *Enterprise Interoperability*, p. 208–212, 2015.

[102] G. Macé-Ramète, J. Lamothe, M. Lauras et F. Bénaben : A road crisis management metamodel for an information decision support system. *Dans 6th IEEE International Conference on Digital Ecosystems Technologies*, 2012.

[103] J.-L. Marichal : *Aggregation operators for multicriteria decision aid.* Thèse de doctorat, University of Liege, 1998.

[104] B. Mayag, M. Grabisch et C. Labreuche : A representation of preferences by the Choquet integral with respect to a 2-additive capacity. *Theory and Decision*, 71(3):297–324, 2011.

[105] B. Mayag, G. Michel et C. Labreuche : Dealing with inconsistencies in the representation of ordinal information by a 2-additive Choquet integral. *Cahier d'Etudes et de Recherche - Laboratoire Génie Industriel, Ecole Centrale Paris*, 11(7):9, 2011.

[106] J. McCarthy : Situations, Actions and Causal Laws. Rap. tech., DTIC Document, 1963.

[107] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld et D. Wilkins : PDDL - The Planning Domain Definition Language. Rap. tech., CVC TR-98-003/DCS TR-1165, 1998.

[108] U. Montanari : Networks of constraints : Fundamental properties and applications to picture processing. *Information sciences*, 7:95–132, 1974.

[109] A. Montarnal, A.-M. Delanoë, F. Bénaben, M. Lauras et J. Lamothe : A PaaS to support collaborations through service composition. *Dans IEEE International Conference on Services Computing (SSC)*, p. 677–684, 2014.

[110] J. Montmain, C. Labreuche, A. Imoussaten et F. Trousset : Multicriteria improvement of complex systems. *Information Sciences*, 291:61–84, 2015.

[111] W. Mu, F. Bénaben, H. Pingaud, N. Boissel-Dallier et J.-P. Lorré : A model-driven BPM approach for SOA mediation information system design in a collaborative context. *Dans IEEE International Conference on Services Computing (SSC)*, p. 747–748, 2011.

[112] Object Management Group : Business Process Model and Notation (BPMN) Version 2.0. Rap. tech., Object Management Group, 2011.

[113] J. Pearl : Heuristics : intelligent search strategies for computer problem solving. 1984.

[114] E. Pednault : ADL : Exploring the Middle Ground Between STRIPS and the Situation Calculus. *Dans Proceedings of the First International*

Conference on Principles of Knowledge Representation and Reasoning, p. 324–332, 1989.

[115] D. Pellier : PDDL4J V2.0.0. Jan. 2015, DOI 10.5281/zenodo.13921, http ://dx.doi.org/10.5281/zenodo.13921.

[116] J.-P. Pignon et C. Labreuche : A methodological approach for operational and technical experimentation based evaluation of systems of systems architectures. *Dans International Conference on Software & Systems Engineering and their Applications (ICSSEA)*, 2007.

[117] G. Pigozzi, A. Tsoukiàs et P. Viappiani : Preferences in Artificial Intelligence. *Annals of Mathematics and Artificial Intelligence*, p. 1–41, 2005.

[118] J.-C. Pomerol et S. Barba-Romero : *Multicriterion Decision in Management : Principles and Practice*, vol. 25. Springer, 2012.

[119] V. Rajsiri, J.-P. Lorré, F. Bénaben et H. Pingaud : Knowledge-based system for collaborative process specification. *Computer in Industry*, 61(2): 161–175, 2010.

[120] R. Reiter : *Knowledge in Action : Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.

[121] F. Rossi, K. B. Venable et T. Walsh : Preferences in Constraint Satisfaction and Optimization. *AI Magazine*, 29(4):58, 2009.

[122] G.-C. Rota : On the Foundations of Combinatorial Theory I. Theory of Möbius Functions. *Probability Theory and Related Fields*, 2(4):340–368, 1964.

[123] B. Roy : Paradigms and Challenges. *Dans Multiple Criteria Decision Analysis : State of the art surveys*, p. 3–24. Springer, 2005.

[124] B. Roy et D. Bouyssou : *Aide Multicritère à la Décision : Méthodes et Cas*. Editions Economica, 1993.

[125] M. Sadeghi, K. Hadj-Hamou et F. Noel : A collaborative platform architecture for coherence management in multi-view integrated product modeling. *International Journal of Computer Integrated Manufacturing*, 23(3):270 – 282, 2010.

[126] L. S. Shapley : A value for n-person games. *Dans Contributions to the Theory of Games, vol. II*, p. 307–317. Princeton University Press, 1953.

[127] I. Singer : Extensions of functions of 0-1 variables to combinatorial optimization. *Numerical Functional Analysis and Optimization*, 7(1):23–62, 1985.

[128] A. Sirko, S. Truptil, A.-M. Barthe-Delanoë et F. Bénaben : Analysing Internet of Things to Feed Internet of Knowledge. *Enterprise Interoperability*, p. 325–330, 2015.

[129] P. Slovic, M. L. Finucane, E. Peters et D. G. MacGregor : The affect heuristic. *European Journal of Operational Research*, 177(3):1333–1352, 2007.

[130] D. Smith : Choosing Objectives in Over-Subscription Planning. *Dans Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS)*, vol. 4, p. 393, 2004.

[131] T. C. Son et E. Pontelli : Planning with Preferences Using Logic Programming. *Theory and Practice of Logic Programming*, 6(5):559–607, 2006.

[132] M. Sugeno : *Theory of fuzzy integrals and its applications*. Thèse de doctorat, Tokyo Institute of Technology, 1974.

[133] W. Thomas : Automata on infinite objects. *Handbook of theoretical computer science, Volume B*, p. 133–191, 1990.

[134] J. Touzi : *Aide à la conception de Système d'Information Collaboratif support de l'interopérabilité des entreprises*. Thèse de doctorat, Institut National Polytechnique de Toulouse, 2007.

[135] W. Treurniet, K. Van Buul-Besseling et J. Wolbers : Collaboration Awareness. A necessity in crisis response coordination. *Dans* 9^{th} *International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, 2012.

[136] S. Truptil : *Etude de l'approche de l'interopérabilité par médiation dans le cadre d'une dynamique de collaboration appliquée à la gestion de crise*. Thèse de doctorat, Institut National Polytechnique de Toulouse, 2011.

[137] S. Truptil, F. Bénaben, V. Chapurlat, C. Hanachi, P. J.-P. et N. Salatge : Démarche de création d'un processus collaboratif de réponse à une crise. *Dans Workshop interdisciplinaire sur la sécurité globale (WISG)*, 2009.

[138] S. Truptil, F. Bénaben, P. Couget, M. Lauras, V. Chapurlat et H. Pingaud : Interoperability of information systems in crisis management :

Crisis modeling and metamodeling. *Dans Enterprise Interoperability III*, p. 583–594. Springer, 2008.

[139] S. Truptil, F. Bénaben et H. Pingaud : Collaborative process design for Mediation Information System Engineering. *Dans 6th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, 2009.

[140] S. Truptil, F. Bénaben, N. Salatge, C. Hanachi, V. Chapurlat, J.-P. Pignon et H. Pingaud : Mediation Information System Engineering for Interoperability Support in Crisis Management. *Dans Enterprise Interoperability IV*, p. 187–197. Springer, 2010.

[141] O. S. Vaidya et S. Kumar : Analytic hierarchy process : An overview of applications. *European Journal of Operational Research*, 169(1):1–29, 2006.

[142] B. Van De Walle et M. Turoff : Emergency Response Information Systems : Emerging Trends and Technologies. *Communications of the ACM*, 50(3):29–31, 2007.

[143] B. Van De Walle et M. Turoff : Decision support for emergency situations. *Information Systems and e-Business Management*, 6(3):295–316, 2008.

[144] M. Van Den Briel, S. Romeo, M. B. Do et S. Kambhampati : Effective Approaches for Partial Satisfaction (Oversubscription) Planning. *Dans AAAI*, p. 562–569, 2004.

[145] D. Vanderpooten : The interactive approach in MCDA : A technical framework and some basic conceptions. *Mathematical and Computer Modelling*, 12(10):1213–1220, 1989.

[146] J. Vincke et P. Brans : A Preference Ranking Organisation Method : The PROMETHEE Method for MCDM. *Management Science*, 31(6):647–656, 1985.

[147] J. Wallenius, J. S. Dyer, P. C. Fishburn, R. Steuer, S. Zionts et K. Deb : Multiple Criteria Decision Making, Multiattribute Utility Theory : Recent Accomplishements and What Lies Ahead. *Management Science*, 54(7):1336–1349, 2008.

[148] R. R. Yager : On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on Systems on Man and Cybernetics*, 18(1):183–190, 1988.

[149] S. Zribi, F. Bénaben, J.-P. Lorré et H. Pingaud : Enhancing services selection by using non-functional properties within BPMN in SOA context. *Dans Collaborative Systems for Reindustrialization*, p. 305–313. Springer, 2013.

Annexe A

Syntaxe de la logique du premier ordre

Cette annexe définit la syntaxe de la logique du premier ordre à savoir les symboles et règles de grammaire utilisés pour construire des formules valides.

Définition A.1 - Symboles de la logique du premier ordre [37]
La syntaxe de la logique du premier ordre inclue des :

- *Symboles de ponctuation* : (et)
- *Connecteurs logiques* : $\neg$ (non), $\wedge$ (et), $\vee$ (ou), $\rightarrow$ (implique)
- *Quantificateurs* : $\forall$ (quel que soit), $\exists$ (il existe)
- *Symboles de variables* : x, y, z, ...
- *Symboles de constantes* : a, b, c, ...
- *Symboles de prédicats* : P, Q, R, ...
- *Symboles de fonctions* : f_1, f_2, f_3, ...

Les symboles de prédicats et de fonctions possèdent une *arité* qui précise le nombre de variables qu'ils manipulent.

Les prédicats sont utilisés pour dénoter des propriétés d'objets ou des relations entre objets. Les variables propositionnelles du calcul des propositions correspondent donc aux prédicats d'arité zéro de la logique du premier ordre.

Les symboles de fonctions dénotent quant à eux des fonctions d'un ensemble d'objets vers un objet. Par conséquent, les fonctions d'arité zéro correspondent aux constantes.

Définition A.2 - Grammaire de la logique du premier ordre [37]
L'ensemble des *termes* est défini par induction par les règles suivantes :

1. *Variables* : Si x est un symbole de variable, alors x est un terme.
2. *Fonctions* : Si f est un symbole de fonction d'arité n et que $t_1, \cdots, t_n$ sont des termes, alors $f(t_1, \cdots, t_n)$ est un terme. En particulier, si c est un symbole de constante, alors c est un terme.

L'ensemble des *formules* est défini par induction par les règles suivantes :

1. *Prédicats* : Si P est un symbole de prédicat d'arité n et que $t_1, \cdots, t_n$ sont des termes, alors $P(t_1, \cdots, t_n)$ est une formule.
2. *Négation* : Si ϕ est une formule alors $\neg\phi$ est une formule.
3. *Connecteurs binaires* : Si ϕ et ψ sont des formules, alors $\phi \wedge \psi$, $\phi \vee \psi$ et $\phi \rightarrow \psi$ sont des formules.
4. *Quantificateurs* : Si ϕ est une formule et x est une variable, alors $\forall x\phi$ et $\exists x\phi$ sont des formules.

Les *atomes* (ou *formules atomiques*) sont les formules obtenues uniquement en utilisant la règle de construction 1. Les *littéraux* sont les formules obtenues uniquement en utilisant les règles de construction 1 et 2. En outre, une formule est dite *close* si elle ne contient pas de symboles de variables.

Annexe B

Problèmes de planification « Rovers »

Cette annexe présente la formalisation PDDL du problème *Rovers* [42] qui a été introduit en 2002 lors de la troisième compétition internationale de planification (IPC3). Il met en œuvre des robots qui réalisent une exploration planétaire. Ces derniers doivent prendre des photographies et récolter des échantillons de sols ou de roches de plusieurs lieux différents.

Les sections B.1 à B.4 présentent les versions simplifiées du problème *Rovers* qui ont été utilisées à des fins d'illustration tout au long de ce manuscrit. La section B.5 présente quant à elle un problème utilisé lors des compétitions internationales de planification à savoir le $18^{ème}$ problème *Rovers* de type « Qualitative Preferences ».

B.1 *Rovers - Exemple (simple)*

domain1.pddl

```
(define (domain Rover)
  (:requirements :strips :typing)
  (:types robot lieu)

  (:predicates
    (presence_echantillon_sol ?x - lieu)
    (presence_echantillon_roche ?x - lieu)
    (visible ?x - lieu ?y - lieu)
    (accessible ?x - lieu ?y - lieu)
    (position ?x - robot ?y - lieu)
```

```
    (possede_echantillon_sol ?x - robot ?y - lieu)
    (possede_echantillon_roche ?x - robot ?y - lieu)
    (possede_photographie ?x - robot ?y - lieu)
  )

  (:action Naviguer
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (accessible ?y ?z))
    :effect (and
      (not (position ?x ?y))
      (position ?x ?z))
  )

  (:action Prelever_Echantillon_Sol
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_sol ?y))
    :effect (and
      (possede_echantillon_sol ?x ?y))
  )

  (:action Prelever_Echantillon_Roche
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_roche ?y))
    :effect (and
      (possede_echantillon_roche ?x ?y))
  )

  (:action Photographier
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (visible ?y ?z))
    :effect (and
      (possede_photographie ?x ?z))
  )
)
```

problem1.pddl

```
(define (problem rover-simple-1)
  (:domain Rover)

  (:objects
    N1 - robot
    L1 - lieu L2 - lieu L3 - lieu
    L4 - lieu L5 - lieu L6 - lieu
  )
```

```
  (:init
    (presence_echantillon_sol L1)
    (presence_echantillon_sol L2)
    (presence_echantillon_sol L3)
    (presence_echantillon_sol L4)
    (presence_echantillon_sol L5)
    (presence_echantillon_sol L6)
    (presence_echantillon_roche L2)
    (presence_echantillon_roche L3)
    (presence_echantillon_roche L6)
    (visible L1 L2) (visible L2 L1)
    (visible L1 L4) (visible L4 L1)
    (visible L1 L5) (visible L5 L1)
    (visible L2 L3) (visible L3 L2)
    (visible L2 L4) (visible L4 L2)
    (visible L2 L5) (visible L5 L2)
    (visible L3 L5) (visible L5 L3)
    (visible L3 L6) (visible L6 L3)
    (visible L4 L5) (visible L5 L4)
    (visible L5 L6) (visible L6 L5)
    (accessible L1 L5) (accessible L5 L1)
    (accessible L2 L4) (accessible L4 L2)
    (accessible L2 L5) (accessible L5 L2)
    (accessible L3 L5) (accessible L5 L3)
    (accessible L3 L6) (accessible L6 L3)
    (accessible L4 L5) (accessible L5 L4)
    (accessible L5 L6) (accessible L6 L5)
    (position N1 L5)
  )

  (:goal (and
    (possede_echantillon_sol N1 L4)
    (possede_echantillon_roche N1 L3)
    (possede_photographie N1 L1)
  )
)
```

B.2 *Rovers - Exemple (expressions numériques)*

domain2.pddl

```
(define (domain Rover)
  (:requirements :strips :typing :numeric-fluents)
  (:types robot lieu)

  (:functions
    (energy ?x - robot) - number
  )
```

```
  (:predicates
    (presence_echantillon_sol ?x - lieu)
    (presence_echantillon_roche ?x - lieu)
    (visible ?x - lieu ?y - lieu)
    (accessible ?x - lieu ?y - lieu)
    (position ?x - robot ?y - lieu)
    (possede_echantillon_sol ?x - robot ?y - lieu)
    (possede_echantillon_roche ?x - robot ?y - lieu)
    (possede_photographie ?x - robot ?y - lieu)
  )

  (:action Naviguer
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (accessible ?y ?z))
      (<= (quantity ?x) 75)
    :effect (and
      (not (position ?x ?y))
      (position ?x ?z))
      (assign (quantity ?x) (+ (quantity ?x) 10))
  )

  (:action Prelever_Echantillon_Sol
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_sol ?y))
    :effect (and
      (possede_echantillon_sol ?x ?y))
  )

  (:action Prelever_Echantillon_Roche
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_roche ?y))
    :effect (and
      (possede_echantillon_roche ?x ?y))
  )

  (:action Photographier
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (visible ?y ?z))
    :effect (and
      (possede_photographie ?x ?z))
  )
)
```

problem2.pddl

```
(define (problem rover-simple-1)
  (:domain Rover)

  (:objects
    N1 - robot
    L1 - lieu L2 - lieu L3 - lieu
    L4 - lieu L5 - lieu L6 - lieu
  )

  (:init
    (presence_echantillon_sol L1)
    (presence_echantillon_sol L2)
    (presence_echantillon_sol L3)
    (presence_echantillon_sol L4)
    (presence_echantillon_sol L5)
    (presence_echantillon_sol L6)
    (presence_echantillon_roche L2)
    (presence_echantillon_roche L3)
    (presence_echantillon_roche L6)
    (visible L1 L2) (visible L2 L1)
    (visible L1 L4) (visible L4 L1)
    (visible L1 L5) (visible L5 L1)
    (visible L2 L3) (visible L3 L2)
    (visible L2 L4) (visible L4 L2)
    (visible L2 L5) (visible L5 L2)
    (visible L3 L5) (visible L5 L3)
    (visible L3 L6) (visible L6 L3)
    (visible L4 L5) (visible L5 L4)
    (visible L5 L6) (visible L6 L5)
    (accessible L1 L5) (accessible L5 L1)
    (accessible L2 L4) (accessible L4 L2)
    (accessible L2 L5) (accessible L5 L2)
    (accessible L3 L5) (accessible L5 L3)
    (accessible L3 L6) (accessible L6 L3)
    (accessible L4 L5) (accessible L5 L4)
    (accessible L5 L6) (accessible L6 L5)
    (position N1 L5)
  )

  (:goal (and
    (possede_echantillon_sol N1 L4)
    (possede_echantillon_roche N1 L3)
    (possede_photographie N1 L1)
  )
)
```

B.3 *Rovers - Exemple (préférences PDDL3)*

domain3.pddl

```
(define (domain Rover)
  (:requirements :strips :typing)
  (:types robot lieu)

  (:functions
    (energy ?x - robot) - number
  )

  (:predicates
    (presence_echantillon_sol ?x - lieu)
    (presence_echantillon_roche ?x - lieu)
    (visible ?x - lieu ?y - lieu)
    (accessible ?x - lieu ?y - lieu)
    (position ?x - robot ?y - lieu)
    (possede_echantillon_sol ?x - robot ?y - lieu)
    (possede_echantillon_roche ?x - robot ?y - lieu)
    (possede_photographie ?x - robot ?y - lieu)
  )

  (:action Naviguer
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (accessible ?y ?z))
      (<= (quantity ?x) 75)
    :effect (and
      (not (position ?x ?y))
      (position ?x ?z))
      (assign (quantity ?x) (+ (quantity ?x) 10))
  )

  (:action Prelever_Echantillon_Sol
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_sol ?y))
    :effect (and
      (possede_echantillon_sol ?x ?y))
  )

  (:action Prelever_Echantillon_Roche
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_roche ?y))
    :effect (and
      (possede_echantillon_roche ?x ?y))
  )
```

```
  (:action Photographier
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (visible ?y ?z))
    :effect (and
      (possede_photographie ?x ?z))
  )
)
```

problem3.pddl

```
(define (problem rover-simple-1)
  (:domain Rover)

  (:objects
    N1 - robot
    L1 - lieu L2 - lieu L3 - lieu
    L4 - lieu L5 - lieu L6 - lieu
  )

  (:init
    (presence_echantillon_sol L1)
    (presence_echantillon_sol L2)
    (presence_echantillon_sol L3)
    (presence_echantillon_sol L4)
    (presence_echantillon_sol L5)
    (presence_echantillon_sol L6)
    (presence_echantillon_roche L2)
    (presence_echantillon_roche L3)
    (presence_echantillon_roche L6)
    (visible L1 L2) (visible L2 L1)
    (visible L1 L4) (visible L4 L1)
    (visible L1 L5) (visible L5 L1)
    (visible L2 L3) (visible L3 L2)
    (visible L2 L4) (visible L4 L2)
    (visible L2 L5) (visible L5 L2)
    (visible L3 L5) (visible L5 L3)
    (visible L3 L6) (visible L6 L3)
    (visible L4 L5) (visible L5 L4)
    (visible L5 L6) (visible L6 L5)
    (accessible L1 L5) (accessible L5 L1)
    (accessible L2 L4) (accessible L4 L2)
    (accessible L2 L5) (accessible L5 L2)
    (accessible L3 L5) (accessible L5 L3)
    (accessible L3 L6) (accessible L6 L3)
    (accessible L4 L5) (accessible L5 L4)
    (accessible L5 L6) (accessible L6 L5)
    (position N1 L5)
  )

  (:goal (and
    (possede_echantillon_sol N1 L4)
```

```
    (possede_echantillon_roche N1 L3)
    (possede_photographie N1 L1)
    (preference f1 (possede_echantillon_sol N1 L6))
    (preference f2 (possede_echantillon_roche N1 L2))
  ))

  (:contraints (and
    (sometime-before (possede_echantillon_roche N1 L3)
      (possede_echantillon_sol N1 L4))
    (preference s1 (sometime (position N1 L1)))
    (preference a1 (at-most-once (position N1 L6)))
  ))

  (:metric minimize (+
    (* (is-violated f1) 80)
    (* (is-violated f2) 80)
    (* (is-violated s1) 100)
    (* (is-violated a1) 80)
    (energy N1)
  ))
)
```

B.4 *Rovers - Exemple (préférences PDDL3/MAUT)*

domain4.pddl

```
(define (domain Rover)
  (:requirements :strips :typing)
  (:types robot lieu)

  (:functions
    (energy ?x - robot) - number
  )

  (:predicates
    (presence_echantillon_sol ?x - lieu)
    (presence_echantillon_roche ?x - lieu)
    (visible ?x - lieu ?y - lieu)
    (accessible ?x - lieu ?y - lieu)
    (position ?x - robot ?y - lieu)
    (possede_echantillon_sol ?x - robot ?y - lieu)
    (possede_echantillon_roche ?x - robot ?y - lieu)
    (possede_photographie ?x - robot ?y - lieu)
  )

  (:action Naviguer
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (accessible ?y ?z))
```

```
      (<= (quantity ?x) 75)
    :effect (and
      (not (position ?x ?y))
      (position ?x ?z))
      (assign (quantity ?x) (+ (quantity ?x) 10))
  )

  (:action Prelever_Echantillon_Sol
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_sol ?y))
    :effect (and
      (possede_echantillon_sol ?x ?y))
  )

  (:action Prelever_Echantillon_Roche
    :parameters (?x - robot ?y - lieu)
    :precondition (and
      (position ?x ?y)
      (presence_echantillon_roche ?y))
    :effect (and
      (possede_echantillon_roche ?x ?y))
  )

  (:action Photographier
    :parameters (?x - robot ?y - lieu ?z - lieu)
    :precondition (and
      (position ?x ?y)
      (visible ?y ?z))
    :effect (and
      (possede_photographie ?x ?z))
  )
)
```

problem4.pddl

```
(define (problem rover-simple-1)
  (:domain Rover)

  (:objects
    N1 - robot
    L1 - lieu L2 - lieu L3 - lieu
    L4 - lieu L5 - lieu L6 - lieu
  )

  (:init
    (presence_echantillon_sol L1)
    (presence_echantillon_sol L2)
    (presence_echantillon_sol L3)
    (presence_echantillon_sol L4)
    (presence_echantillon_sol L5)
```

```
  (presence_echantillon_sol L6)
  (presence_echantillon_roche L2)
  (presence_echantillon_roche L3)
  (presence_echantillon_roche L6)
  (visible L1 L2) (visible L2 L1)
  (visible L1 L4) (visible L4 L1)
  (visible L1 L5) (visible L5 L1)
  (visible L2 L3) (visible L3 L2)
  (visible L2 L4) (visible L4 L2)
  (visible L2 L5) (visible L5 L2)
  (visible L3 L5) (visible L5 L3)
  (visible L3 L6) (visible L6 L3)
  (visible L4 L5) (visible L5 L4)
  (visible L5 L6) (visible L6 L5)
  (accessible L1 L5) (accessible L5 L1)
  (accessible L2 L4) (accessible L4 L2)
  (accessible L2 L5) (accessible L5 L2)
  (accessible L3 L5) (accessible L5 L3)
  (accessible L3 L6) (accessible L6 L3)
  (accessible L4 L5) (accessible L5 L4)
  (accessible L5 L6) (accessible L6 L5)
  (position N1 L5)
)

(:goal (and
  (possede_echantillon_sol N1 L4)
  (possede_echantillon_roche N1 L3)
  (possede_photographie N1 L1)
  (preference f1 (possede_echantillon_sol N1 L6))
  (preference f2 (possede_echantillon_roche N1 L2))
))

(:contraints (and
  (sometime-before (possede_echantillon_roche N1 L3)
    (possede_echantillon_sol N1 L4))
  (preference s1 (sometime (position N1 L1)))
  (preference a1 (at-most-once (position N1 L6)))
))

(:maut-preferences
  (:numeric-criterion c-e1
    :attribute (energy N1)
    :utility-function (
      (40, 1)
      (80, 0.6)
      (100, 0)
    )
  )

  (:trajectory-criterion c-f1
    :preference (f1))

  (:trajectory-criterion c-s1
    :preference (s1))
```

```
    (:trajectory-criterion c-a1
      :preference (a1))

    (:choquet-integral choqInt
      :mobius (
        (c-e1 0.4)
        (c-f1 0.25)
        (c-s1 0.3)
        (c-a1 0.3)
        (c-e1 c-f1 -0.2)
        (c-s1 c-a1 0.15)
      ))
    )
  )

  (:metric maximize choqInt)
)
```

B.5 *Rovers - Qualitative Preferences (IPC)*

domain5.pddl

```
(define (domain Rover)
  (:requirements :typing :constraints :preferences)
  (:types rover waypoint store camera mode lander objective)

  (:predicates
    (at ?x - rover ?y - waypoint)
    (at_lander ?x - lander ?y - waypoint)
    (can_traverse ?r - rover ?x - waypoint ?y - waypoint)
    (equipped_for_soil_analysis ?r - rover)
    (equipped_for_rock_analysis ?r - rover)
    (equipped_for_imaging ?r - rover)
    (empty ?s - store)
    (have_rock_analysis ?r - rover ?w - waypoint)
    (have_soil_analysis ?r - rover ?w - waypoint)
    (full ?s - store)
    (calibrated ?c - camera ?r - rover)
    (supports ?c - camera ?m - mode)
    (available ?r - rover)
    (visible ?w - waypoint ?p - waypoint)
    (have_image ?r - rover ?o - objective ?m - mode)
    (communicated_soil_data ?w - waypoint)
    (communicated_rock_data ?w - waypoint)
    (communicated_image_data ?o - objective ?m - mode)
    (at_soil_sample ?w - waypoint)
    (at_rock_sample ?w - waypoint)
    (visible_from ?o - objective ?w - waypoint)
    (store_of ?s - store ?r - rover)
```

```
    (calibration_target ?i - camera ?o - objective)
    (on_board ?i - camera ?r - rover)
    (channel_free ?l - lander)
  )

  (:action navigate
    :parameters (?x - rover ?y - waypoint ?z - waypoint)
    :precondition (and
      (can_traverse ?x ?y ?z)
      (available ?x)
      (at ?x ?y)
      (visible ?y ?z)
    )
    :effect (and
      (not (at ?x ?y))
      (at ?x ?z)
    )
  )

  (:action sample_soil
    :parameters (?x - rover ?s - store ?p - waypoint)
    :precondition (and
      (at ?x ?p)
      (at_soil_sample ?p)
      (equipped_for_soil_analysis ?x)
      (store_of ?s ?x)
      (empty ?s)
    )
    :effect (and
      (not (empty ?s))
      (full ?s)
      (have_soil_analysis ?x ?p)
      (not (at_soil_sample ?p))
    )
  )

  (:action sample_rock
    :parameters (?x - rover ?s - store ?p - waypoint)
    :precondition (and
      (at ?x ?p)
      (at_rock_sample ?p)
      (equipped_for_rock_analysis ?x)
      (store_of ?s ?x)
      (empty ?s)
    )
    :effect (and
      (not (empty ?s))
      (full ?s)
      (have_rock_analysis ?x ?p)
      (not (at_rock_sample ?p))
    )
  )
```

```
(:action drop
  :parameters (?x - rover ?y - store)
  :precondition (and
    (store_of ?y ?x)
    (full ?y)
  )
  :effect (and
    (not (full ?y))
    (empty ?y)
  )
)

(:action calibrate
  :parameters (?r - rover ?i - camera ?t - objective ?w - waypoint)
  :precondition (and
    (equipped_for_imaging ?r)
    (calibration_target ?i ?t)
    (at ?r ?w)
    (visible_from ?t ?w)
    (on_board ?i ?r)
  )
  :effect (calibrated ?i ?r)
)

(:action take_image
  :parameters (?r - rover ?p - waypoint ?o - objective ?i - camera ?m - mode)
  :precondition (and
    (calibrated ?i ?r)
    (on_board ?i ?r)
    (equipped_for_imaging ?r)
    (supports ?i ?m)
    (visible_from ?o ?p)
    (at ?r ?p)
  )
  :effect (and
    (have_image ?r ?o ?m)
    (not (calibrated ?i ?r))
  )
)

(:action communicate_soil_data
  :parameters (?r - rover ?l - lander ?p - waypoint ?x - waypoint ?y - waypoint)
  :precondition (and
    (at ?r ?x)
    (at_lander ?l ?y)
    (have_soil_analysis ?r ?p)
    (visible ?x ?y)
    (available ?r)
    (channel_free ?l)
  )
  :effect (communicated_soil_data ?p)
)
```

```
  (:action communicate_rock_data
    :parameters (?r - rover ?l - lander ?p - waypoint ?x - waypoint ?y - waypoint)
    :precondition (and
      (at ?r ?x)
      (at_lander ?l ?y)
      (have_rock_analysis ?r ?p)
      (visible ?x ?y)
      (available ?r)
      (channel_free ?l)
    )
    :effect (communicated_rock_data ?p)
  )

  (:action communicate_image_data
    :parameters (?r - rover ?l - lander ?o - objective
      ?m - mode ?x - waypoint ?y - waypoint)
    :precondition (and
      (at ?r ?x)
      (at_lander ?l ?y)
      (have_image ?r ?o ?m)
      (visible ?x ?y)
      (available ?r)
      (channel_free ?l)
    )
    :effect (communicated_image_data ?o ?m)
  )
)
```

problem5.pddl

```
(define (problem roverprob4621)
  (:domain rover)
  (:objects general - lander colour - mode high_res - mode
    low_res - mode rover0 - rover rover1 - rover rover2 - rover
    rover3 - rover rover4 - rover rover5 - rover rover0store - store
    rover1store - store rover2store - store rover3store - store
    rover4store - store rover5store - store waypoint0 - waypoint
    waypoint1 - waypoint waypoint2 - waypoint waypoint3 - waypoint
    waypoint4 - waypoint waypoint5 - waypoint waypoint6 - waypoint
    waypoint7 - waypoint waypoint8 - waypoint waypoint9 - waypoint
    waypoint10 - waypoint waypoint11 - waypoint waypoint12 - waypoint
    waypoint13 - waypoint waypoint14 - waypoint waypoint15 - waypoint
    waypoint16 - waypoint waypoint17 - waypoint waypoint18 - waypoint
    waypoint19 - waypoint camera0 - camera camera1 - camera
    camera2 - camera camera3 - camera camera4 - camera camera5 - camera
    camera6 - camera objective0 - objective objective1 - objective
    objective2 - objective objective3 - objective objective4 - objective
    objective5 - objective objective6 - objective)

  (:init
    (visible waypoint0 waypoint1) (visible waypoint1 waypoint0)
    (visible waypoint0 waypoint2) (visible waypoint2 waypoint0)
```

```
(visible waypoint0 waypoint12) (visible waypoint12 waypoint0)
(visible waypoint0 waypoint13) (visible waypoint13 waypoint0)
(visible waypoint1 waypoint2) (visible waypoint2 waypoint1)
(visible waypoint1 waypoint6) (visible waypoint6 waypoint1)
(visible waypoint1 waypoint11) (visible waypoint11 waypoint1)
(visible waypoint1 waypoint15) (visible waypoint15 waypoint1)
(visible waypoint2 waypoint6) (visible waypoint6 waypoint2)
(visible waypoint2 waypoint9) (visible waypoint9 waypoint2)
(visible waypoint2 waypoint14) (visible waypoint14 waypoint2)
(visible waypoint2 waypoint16) (visible waypoint16 waypoint2)
(visible waypoint2 waypoint18) (visible waypoint18 waypoint2)
(visible waypoint3 waypoint7) (visible waypoint7 waypoint3)
(visible waypoint3 waypoint11) (visible waypoint11 waypoint3)
(visible waypoint3 waypoint13) (visible waypoint13 waypoint3)
(visible waypoint3 waypoint15) (visible waypoint15 waypoint3)
(visible waypoint4 waypoint5) (visible waypoint5 waypoint4)
(visible waypoint4 waypoint9) (visible waypoint9 waypoint4)
(visible waypoint4 waypoint12) (visible waypoint12 waypoint4)
(visible waypoint4 waypoint18) (visible waypoint18 waypoint4)
(visible waypoint5 waypoint6) (visible waypoint6 waypoint5)
(visible waypoint5 waypoint10) (visible waypoint10 waypoint5)
(visible waypoint5 waypoint12) (visible waypoint12 waypoint5)
(visible waypoint5 waypoint14) (visible waypoint14 waypoint5)
(visible waypoint6 waypoint3) (visible waypoint3 waypoint6)
(visible waypoint6 waypoint12) (visible waypoint12 waypoint6)
(visible waypoint6 waypoint13) (visible waypoint13 waypoint6)
(visible waypoint6 waypoint14) (visible waypoint14 waypoint6)
(visible waypoint6 waypoint19) (visible waypoint19 waypoint6)
(visible waypoint7 waypoint1) (visible waypoint1 waypoint7)
(visible waypoint7 waypoint2) (visible waypoint2 waypoint7)
(visible waypoint7 waypoint5) (visible waypoint5 waypoint7)
(visible waypoint7 waypoint9) (visible waypoint9 waypoint7)
(visible waypoint8 waypoint14) (visible waypoint14 waypoint8)
(visible waypoint9 waypoint0) (visible waypoint0 waypoint9)
(visible waypoint9 waypoint3) (visible waypoint3 waypoint9)
(visible waypoint9 waypoint5) (visible waypoint5 waypoint9)
(visible waypoint9 waypoint6) (visible waypoint6 waypoint9)
(visible waypoint9 waypoint8) (visible waypoint8 waypoint9)
(visible waypoint9 waypoint12) (visible waypoint12 waypoint9)
(visible waypoint10 waypoint3) (visible waypoint3 waypoint10)
(visible waypoint10 waypoint7) (visible waypoint7 waypoint10)
(visible waypoint10 waypoint8) (visible waypoint8 waypoint10)
(visible waypoint10 waypoint9) (visible waypoint9 waypoint10)
(visible waypoint10 waypoint16) (visible waypoint16 waypoint10)
(visible waypoint11 waypoint4) (visible waypoint4 waypoint11)
(visible waypoint11 waypoint5) (visible waypoint5 waypoint11)
(visible waypoint11 waypoint7) (visible waypoint7 waypoint11)
(visible waypoint11 waypoint12) (visible waypoint12 waypoint11)
(visible waypoint11 waypoint16) (visible waypoint16 waypoint11)
(visible waypoint12 waypoint1) (visible waypoint1 waypoint12)
(visible waypoint12 waypoint8) (visible waypoint8 waypoint12)
(visible waypoint12 waypoint14) (visible waypoint14 waypoint12)
(visible waypoint12 waypoint19) (visible waypoint19 waypoint12)
(visible waypoint13 waypoint4) (visible waypoint4 waypoint13)
```

```
(visible waypoint13 waypoint19) (visible waypoint19 waypoint13)
(visible waypoint14 waypoint9) (visible waypoint9 waypoint14)
(visible waypoint14 waypoint15) (visible waypoint15 waypoint14)
(visible waypoint15 waypoint6) (visible waypoint6 waypoint15)
(visible waypoint16 waypoint0) (visible waypoint0 waypoint16)
(visible waypoint16 waypoint12) (visible waypoint12 waypoint16)
(visible waypoint16 waypoint13) (visible waypoint13 waypoint16)
(visible waypoint17 waypoint0) (visible waypoint0 waypoint17)
(visible waypoint17 waypoint11) (visible waypoint11 waypoint17)
(visible waypoint18 waypoint0) (visible waypoint0 waypoint18)
(visible waypoint18 waypoint9) (visible waypoint9 waypoint18)
(visible waypoint18 waypoint11) (visible waypoint11 waypoint18)
(visible waypoint19 waypoint9) (visible waypoint9 waypoint19)
(visible waypoint19 waypoint14) (visible waypoint14 waypoint19)
(visible waypoint19 waypoint15) (visible waypoint15 waypoint19)
(visible waypoint19 waypoint16) (visible waypoint16 waypoint19)
(visible waypoint19 waypoint17) (visible waypoint17 waypoint19)
(at_soil_sample waypoint0) (at_rock_sample waypoint1)
(at_rock_sample waypoint2) (at_soil_sample waypoint3)
(at_rock_sample waypoint4) (at_rock_sample waypoint5)
(at_rock_sample waypoint6) (at_rock_sample waypoint7)
(at_soil_sample waypoint8) (at_soil_sample waypoint9)
(at_rock_sample waypoint9) (at_soil_sample waypoint10)
(at_soil_sample waypoint11) (at_soil_sample waypoint12)
(at_rock_sample waypoint12) (at_soil_sample waypoint13)
(at_rock_sample waypoint13) (at_soil_sample waypoint14)
(at_soil_sample waypoint15) (at_soil_sample waypoint17)
(at_rock_sample waypoint17) (at_soil_sample waypoint18)
(at_rock_sample waypoint18) (at_soil_sample waypoint19)
(at_lander general waypoint17) (channel_free general)
(at rover0 waypoint2) (available rover0) (store_of rover0store rover0)
(empty rover0store) (equipped_for_soil_analysis rover0)
(equipped_for_rock_analysis rover0) (equipped_for_imaging rover0)
(accessible rover0 waypoint2 waypoint0) (accessible rover0 waypoint0 waypoint2)
(accessible rover0 waypoint2 waypoint6) (accessible rover0 waypoint6 waypoint2)
(accessible rover0 waypoint2 waypoint7) (accessible rover0 waypoint7 waypoint2)
(accessible rover0 waypoint2 waypoint9) (accessible rover0 waypoint9 waypoint2)
(accessible rover0 waypoint2 waypoint14) (accessible rover0 waypoint14 waypoint2)
(accessible rover0 waypoint2 waypoint18) (accessible rover0 waypoint18 waypoint2)
(accessible rover0 waypoint0 waypoint12) (accessible rover0 waypoint12 waypoint0)
(accessible rover0 waypoint0 waypoint13) (accessible rover0 waypoint13 waypoint0)
(accessible rover0 waypoint6 waypoint1) (accessible rover0 waypoint1 waypoint6)
(accessible rover0 waypoint6 waypoint15) (accessible rover0 waypoint15 waypoint6)
(accessible rover0 waypoint6 waypoint19) (accessible rover0 waypoint19 waypoint6)
(accessible rover0 waypoint7 waypoint3) (accessible rover0 waypoint3 waypoint7)
(accessible rover0 waypoint7 waypoint11) (accessible rover0 waypoint11 waypoint7)
(accessible rover0 waypoint9 waypoint4) (accessible rover0 waypoint4 waypoint9)
(accessible rover0 waypoint9 waypoint5) (accessible rover0 waypoint5 waypoint9)
(accessible rover0 waypoint9 waypoint8) (accessible rover0 waypoint8 waypoint9)
(accessible rover0 waypoint13 waypoint16) (accessible rover0 waypoint16 waypoint13)
(accessible rover0 waypoint5 waypoint10) (accessible rover0 waypoint10 waypoint5)
(at rover1 waypoint9) (available rover1) (store_of rover1store rover1)
(empty rover1store) (equipped_for_rock_analysis rover1)
(equipped_for_imaging rover1)
```

```
(accessible rover1 waypoint9 waypoint0) (accessible rover1 waypoint0 waypoint9)
(accessible rover1 waypoint9 waypoint3) (accessible rover1 waypoint3 waypoint9)
(accessible rover1 waypoint9 waypoint4) (accessible rover1 waypoint4 waypoint9)
(accessible rover1 waypoint9 waypoint7) (accessible rover1 waypoint7 waypoint9)
(accessible rover1 waypoint9 waypoint10) (accessible rover1 waypoint10 waypoint9)
(accessible rover1 waypoint9 waypoint12) (accessible rover1 waypoint12 waypoint9)
(accessible rover1 waypoint9 waypoint14) (accessible rover1 waypoint14 waypoint9)
(accessible rover1 waypoint9 waypoint18) (accessible rover1 waypoint18 waypoint9)
(accessible rover1 waypoint9 waypoint19) (accessible rover1 waypoint19 waypoint9)
(accessible rover1 waypoint0 waypoint2) (accessible rover1 waypoint2 waypoint0)
(accessible rover1 waypoint0 waypoint13) (accessible rover1 waypoint13 waypoint0)
(accessible rover1 waypoint0 waypoint16) (accessible rover1 waypoint16 waypoint0)
(accessible rover1 waypoint3 waypoint6) (accessible rover1 waypoint6 waypoint3)
(accessible rover1 waypoint4 waypoint11) (accessible rover1 waypoint11 waypoint4)
(accessible rover1 waypoint7 waypoint5) (accessible rover1 waypoint5 waypoint7)
(accessible rover1 waypoint10 waypoint8) (accessible rover1 waypoint8 waypoint10)
(accessible rover1 waypoint12 waypoint1) (accessible rover1 waypoint1 waypoint12)
(accessible rover1 waypoint14 waypoint15) (accessible rover1 waypoint15 waypoint14)
(accessible rover1 waypoint19 waypoint17) (accessible rover1 waypoint17 waypoint19)
(at rover2 waypoint0) (available rover2) (store_of rover2store rover2)
(empty rover2store) (equipped_for_soil_analysis rover2)
(equipped_for_imaging rover2)
(accessible rover2 waypoint0 waypoint1) (accessible rover2 waypoint1 waypoint0)
(accessible rover2 waypoint0 waypoint9) (accessible rover2 waypoint9 waypoint0)
(accessible rover2 waypoint0 waypoint13) (accessible rover2 waypoint13 waypoint0)
(accessible rover2 waypoint0 waypoint16) (accessible rover2 waypoint16 waypoint0)
(accessible rover2 waypoint0 waypoint17) (accessible rover2 waypoint17 waypoint0)
(accessible rover2 waypoint0 waypoint18) (accessible rover2 waypoint18 waypoint0)
(accessible rover2 waypoint1 waypoint2) (accessible rover2 waypoint2 waypoint1)
(accessible rover2 waypoint1 waypoint6) (accessible rover2 waypoint6 waypoint1)
(accessible rover2 waypoint1 waypoint11) (accessible rover2 waypoint11 waypoint1)
(accessible rover2 waypoint1 waypoint12) (accessible rover2 waypoint12 waypoint1)
(accessible rover2 waypoint1 waypoint15) (accessible rover2 waypoint15 waypoint1)
(accessible rover2 waypoint9 waypoint3) (accessible rover2 waypoint3 waypoint9)
(accessible rover2 waypoint9 waypoint4) (accessible rover2 waypoint4 waypoint9)
(accessible rover2 waypoint9 waypoint5) (accessible rover2 waypoint5 waypoint9)
(accessible rover2 waypoint9 waypoint7) (accessible rover2 waypoint7 waypoint9)
(accessible rover2 waypoint9 waypoint10) (accessible rover2 waypoint10 waypoint9)
(accessible rover2 waypoint9 waypoint14) (accessible rover2 waypoint14 waypoint9)
(accessible rover2 waypoint9 waypoint19) (accessible rover2 waypoint19 waypoint9)
(at rover3 waypoint18) (available rover3) (store_of rover3store rover3)
(empty rover3store) (equipped_for_rock_analysis rover3)
(equipped_for_imaging rover3)
(accessible rover3 waypoint18 waypoint0) (accessible rover3 waypoint0 waypoint18)
(accessible rover3 waypoint18 waypoint4) (accessible rover3 waypoint4 waypoint18)
(accessible rover3 waypoint18 waypoint11) (accessible rover3 waypoint11 waypoint18)
(accessible rover3 waypoint0 waypoint1) (accessible rover3 waypoint1 waypoint0)
(accessible rover3 waypoint0 waypoint2) (accessible rover3 waypoint2 waypoint0)
(accessible rover3 waypoint0 waypoint9) (accessible rover3 waypoint9 waypoint0)
(accessible rover3 waypoint0 waypoint12) (accessible rover3 waypoint12 waypoint0)
(accessible rover3 waypoint0 waypoint17) (accessible rover3 waypoint17 waypoint0)
(accessible rover3 waypoint11 waypoint16) (accessible rover3 waypoint16 waypoint11)
(accessible rover3 waypoint1 waypoint6) (accessible rover3 waypoint6 waypoint1)
(accessible rover3 waypoint1 waypoint7) (accessible rover3 waypoint7 waypoint1)
```

```
(accessible rover3 waypoint1 waypoint15) (accessible rover3 waypoint15 waypoint1)
(accessible rover3 waypoint2 waypoint14) (accessible rover3 waypoint14 waypoint2)
(accessible rover3 waypoint9 waypoint3) (accessible rover3 waypoint3 waypoint9)
(accessible rover3 waypoint9 waypoint5) (accessible rover3 waypoint5 waypoint9)
(accessible rover3 waypoint9 waypoint8) (accessible rover3 waypoint8 waypoint9)
(accessible rover3 waypoint12 waypoint19) (accessible rover3 waypoint19 waypoint12)
(accessible rover3 waypoint16 waypoint13) (accessible rover3 waypoint13 waypoint16)
(accessible rover3 waypoint7 waypoint10) (accessible rover3 waypoint10 waypoint7)
(at rover4 waypoint3) (available rover4) (store_of rover4store rover4)
(empty rover4store) (equipped_for_soil_analysis rover4)
(equipped_for_imaging rover4)
(accessible rover4 waypoint3 waypoint6) (accessible rover4 waypoint6 waypoint3)
(accessible rover4 waypoint3 waypoint7) (accessible rover4 waypoint7 waypoint3)
(accessible rover4 waypoint3 waypoint9) (accessible rover4 waypoint9 waypoint3)
(accessible rover4 waypoint3 waypoint10) (accessible rover4 waypoint10 waypoint3)
(accessible rover4 waypoint3 waypoint11) (accessible rover4 waypoint11 waypoint3)
(accessible rover4 waypoint3 waypoint13) (accessible rover4 waypoint13 waypoint3)
(accessible rover4 waypoint3 waypoint15) (accessible rover4 waypoint15 waypoint3)
(accessible rover4 waypoint6 waypoint5) (accessible rover4 waypoint5 waypoint6)
(accessible rover4 waypoint6 waypoint12) (accessible rover4 waypoint12 waypoint6)
(accessible rover4 waypoint6 waypoint14) (accessible rover4 waypoint14 waypoint6)
(accessible rover4 waypoint9 waypoint0) (accessible rover4 waypoint0 waypoint9)
(accessible rover4 waypoint9 waypoint4) (accessible rover4 waypoint4 waypoint9)
(accessible rover4 waypoint9 waypoint8) (accessible rover4 waypoint8 waypoint9)
(accessible rover4 waypoint9 waypoint19) (accessible rover4 waypoint19 waypoint9)
(accessible rover4 waypoint10 waypoint16) (accessible rover4 waypoint16 waypoint10)
(accessible rover4 waypoint11 waypoint17) (accessible rover4 waypoint17 waypoint11)
(accessible rover4 waypoint11 waypoint18) (accessible rover4 waypoint18 waypoint11)
(accessible rover4 waypoint15 waypoint1) (accessible rover4 waypoint1 waypoint15)
(accessible rover4 waypoint14 waypoint2) (accessible rover4 waypoint2 waypoint14)
(at rover5 waypoint0) (available rover5) (store_of rover5store rover5)
(empty rover5store) (equipped_for_rock_analysis rover5)
(equipped_for_imaging rover5)
(accessible rover5 waypoint0 waypoint1) (accessible rover5 waypoint1 waypoint0)
(accessible rover5 waypoint0 waypoint12) (accessible rover5 waypoint12 waypoint0)
(accessible rover5 waypoint0 waypoint13) (accessible rover5 waypoint13 waypoint0)
(accessible rover5 waypoint0 waypoint17) (accessible rover5 waypoint17 waypoint0)
(accessible rover5 waypoint0 waypoint18) (accessible rover5 waypoint18 waypoint0)
(accessible rover5 waypoint1 waypoint2) (accessible rover5 waypoint2 waypoint1)
(accessible rover5 waypoint1 waypoint6) (accessible rover5 waypoint6 waypoint1)
(accessible rover5 waypoint1 waypoint7) (accessible rover5 waypoint7 waypoint1)
(accessible rover5 waypoint1 waypoint11) (accessible rover5 waypoint11 waypoint1)
(accessible rover5 waypoint1 waypoint15) (accessible rover5 waypoint15 waypoint1)
(accessible rover5 waypoint12 waypoint4) (accessible rover5 waypoint4 waypoint12)
(accessible rover5 waypoint12 waypoint5) (accessible rover5 waypoint5 waypoint12)
(accessible rover5 waypoint12 waypoint8) (accessible rover5 waypoint8 waypoint12)
(accessible rover5 waypoint12 waypoint9) (accessible rover5 waypoint9 waypoint12)
(accessible rover5 waypoint12 waypoint14) (accessible rover5 waypoint14 waypoint12)
(accessible rover5 waypoint12 waypoint19) (accessible rover5 waypoint19 waypoint12)
(accessible rover5 waypoint13 waypoint16) (accessible rover5 waypoint16 waypoint13)
(accessible rover5 waypoint7 waypoint3) (accessible rover5 waypoint3 waypoint7)
(accessible rover5 waypoint7 waypoint10) (accessible rover5 waypoint10 waypoint7)
(on_board camera0 rover1) (on_board camera1 rover2) (on_board camera2 rover4)
(on_board camera3 rover3) (on_board camera4 rover3) (on_board camera5 rover0)
```

```
  (on_board camera6 rover5) (supports camera0 high_res) (supports camera1 colour)
  (supports camera1 low_res) (supports camera2 colour) (supports camera2 high_res)
  (supports camera3 colour) (supports camera4 high_res) (supports camera4 low_res)
  (supports camera5 colour) (supports camera6 high_res) (supports camera6 low_res)
  (calibration_target camera0 objective4) (calibration_target camera1 objective6)
  (calibration_target camera2 objective0) (calibration_target camera3 objective6)
  (calibration_target camera4 objective4) (calibration_target camera5 objective2)
  (calibration_target camera6 objective6)
  (visible_from objective0 waypoint0) (visible_from objective0 waypoint1)
  (visible_from objective0 waypoint2) (visible_from objective0 waypoint3)
  (visible_from objective0 waypoint4) (visible_from objective0 waypoint5)
  (visible_from objective0 waypoint6) (visible_from objective0 waypoint7)
  (visible_from objective0 waypoint8) (visible_from objective0 waypoint9)
  (visible_from objective0 waypoint10) (visible_from objective1 waypoint0)
  (visible_from objective1 waypoint1) (visible_from objective1 waypoint2)
  (visible_from objective1 waypoint3) (visible_from objective1 waypoint4)
  (visible_from objective1 waypoint5) (visible_from objective1 waypoint6)
  (visible_from objective1 waypoint7) (visible_from objective1 waypoint8)
  (visible_from objective1 waypoint9) (visible_from objective1 waypoint10)
  (visible_from objective1 waypoint11) (visible_from objective1 waypoint12)
  (visible_from objective2 waypoint0) (visible_from objective2 waypoint1)
  (visible_from objective2 waypoint2) (visible_from objective2 waypoint3)
  (visible_from objective2 waypoint4) (visible_from objective2 waypoint5)
  (visible_from objective2 waypoint6) (visible_from objective2 waypoint7)
  (visible_from objective2 waypoint8) (visible_from objective2 waypoint9)
  (visible_from objective2 waypoint10) (visible_from objective2 waypoint11)
  (visible_from objective2 waypoint12) (visible_from objective2 waypoint13)
  (visible_from objective2 waypoint14) (visible_from objective3 waypoint0)
  (visible_from objective3 waypoint1) (visible_from objective3 waypoint2)
  (visible_from objective3 waypoint3) (visible_from objective3 waypoint4)
  (visible_from objective3 waypoint5) (visible_from objective3 waypoint6)
  (visible_from objective3 waypoint7) (visible_from objective3 waypoint8)
  (visible_from objective3 waypoint9) (visible_from objective3 waypoint10)
  (visible_from objective3 waypoint11) (visible_from objective3 waypoint12)
  (visible_from objective3 waypoint13) (visible_from objective3 waypoint14)
  (visible_from objective3 waypoint15) (visible_from objective3 waypoint16)
  (visible_from objective3 waypoint17) (visible_from objective3 waypoint18)
  (visible_from objective3 waypoint19) (visible_from objective4 waypoint0)
  (visible_from objective4 waypoint1) (visible_from objective4 waypoint2)
  (visible_from objective4 waypoint3) (visible_from objective4 waypoint4)
  (visible_from objective4 waypoint5) (visible_from objective4 waypoint6)
  (visible_from objective4 waypoint7) (visible_from objective4 waypoint8)
  (visible_from objective4 waypoint9) (visible_from objective4 waypoint10)
  (visible_from objective4 waypoint11) (visible_from objective4 waypoint12)
  (visible_from objective4 waypoint13) (visible_from objective4 waypoint14)
  (visible_from objective4 waypoint15) (visible_from objective5 waypoint0)
  (visible_from objective5 waypoint1) (visible_from objective5 waypoint2)
  (visible_from objective6 waypoint0) (visible_from objective6 waypoint1)
  (visible_from objective6 waypoint2) (visible_from objective6 waypoint3)
  (visible_from objective6 waypoint4) (visible_from objective6 waypoint5)
  (visible_from objective6 waypoint6) (visible_from objective6 waypoint7)
  (visible_from objective6 waypoint8) (visible_from objective6 waypoint9)
  (visible_from objective6 waypoint10)
)
```

```
(:goal (and
  (communicated_soil_data waypoint14)
  (communicated_soil_data waypoint0)
  (communicated_rock_data waypoint4)
  (communicated_rock_data waypoint7)
  (communicated_rock_data waypoint2)
  (communicated_rock_data waypoint5)
  (communicated_rock_data waypoint6)
  (communicated_image_data objective5 colour)
  (communicated_image_data objective3 low_res)
  (communicated_image_data objective2 colour)
  (communicated_image_data objective4 high_res)
))

(:constraints
  (and
    (preference a0 (always (at rover4 waypoint3)))
    (preference a1 (always (at rover3 waypoint18)))
    (preference a2 (always (at rover2 waypoint0)))
    (preference a3 (always (empty rover4store)))
    (preference a4 (always (empty rover5store)))
    (preference a5 (always (empty rover3store)))
    (preference e0 (sometime (at rover5 waypoint19)))
    (preference e1 (sometime (at rover5 waypoint12)))
    (preference e2 (sometime (at rover5 waypoint5)))
    (preference e3 (sometime (at rover5 waypoint6)))
    (preference e4 (sometime (at rover4 waypoint11)))
    (preference e5 (sometime (at rover4 waypoint14)))
    (preference e6 (sometime (at rover3 waypoint4)))
    (preference e7 (sometime (at rover3 waypoint7)))
    (preference e8 (sometime (at rover3 waypoint5)))
    (preference e9 (sometime (at rover3 waypoint6)))
    (preference e10 (sometime (at rover2 waypoint19)))
    (preference e11 (sometime (at rover2 waypoint1)))
    (preference e12 (sometime (at rover2 waypoint14)))
    (preference e13 (sometime (at rover2 waypoint11)))
    (preference e14 (sometime (at rover1 waypoint19)))
    (preference e15 (sometime (at rover1 waypoint0)))
    (preference e16 (sometime (at rover1 waypoint11)))
    (preference e17 (sometime (at rover1 waypoint7)))
    (preference e18 (sometime (at rover1 waypoint5)))
    (preference e19 (sometime (at rover0 waypoint19)))
    (preference e20 (sometime (at rover0 waypoint6)))
    (preference e21 (sometime (at rover0 waypoint14)))
    (preference e22 (sometime (at rover0 waypoint11)))
    (preference e23 (sometime (full rover4store)))
    (preference e24 (sometime (have_soil_analysis rover4 waypoint14)))
    (preference e25 (sometime (have_soil_analysis rover2 waypoint14)))
    (preference e26 (sometime (have_soil_analysis rover0 waypoint14)))
    (preference e27 (sometime (have_rock_analysis rover5 waypoint6)))
    (preference e28 (sometime (have_rock_analysis rover5 waypoint5)))
    (preference e29 (sometime (have_rock_analysis rover3 waypoint6)))
    (preference e30 (sometime (have_rock_analysis rover3 waypoint5)))
```

```
(preference e31 (sometime (have_rock_analysis rover3 waypoint4)))
(preference e32 (sometime (have_rock_analysis rover1 waypoint7)))
(preference e33 (sometime (have_rock_analysis rover1 waypoint5)))
(preference e34 (sometime (have_rock_analysis rover1 waypoint4)))
(preference e35 (sometime (have_rock_analysis rover0 waypoint7)))
(preference e36 (sometime (have_rock_analysis rover0 waypoint6)))
(preference e37 (sometime (calibrated camera2 rover4)))
(preference e38 (sometime (calibrated camera4 rover3)))
(preference e39 (sometime (calibrated camera0 rover1)))
(preference e40 (sometime (calibrated camera5 rover0)))
(preference e41 (sometime (have_image rover5 objective3 low_res)))
(preference e42 (sometime (have_image rover5 objective4 high_res)))
(preference e43 (sometime (have_image rover4 objective4 high_res)))
(preference e44 (sometime (have_image rover4 objective2 colour)))
(preference e45 (sometime (have_image rover3 objective3 low_res)))
(preference e46 (sometime (have_image rover3 objective2 colour)))
(preference e47 (sometime (have_image rover3 objective5 colour)))
(preference e48 (sometime (have_image rover2 objective3 low_res)))
(preference e49 (sometime (have_image rover2 objective2 colour)))
(preference e50 (sometime (have_image rover2 objective5 colour)))
(preference e51 (sometime (have_image rover1 objective4 high_res)))
(preference e52 (sometime (have_image rover0 objective2 colour)))
(preference e53 (sometime (have_image rover0 objective5 colour)))
(preference o0 (at-most-once (at rover5 waypoint0)))
(preference o1 (at-most-once (at rover5 waypoint1)))
(preference o2 (at-most-once (at rover5 waypoint11)))
(preference o3 (at-most-once (at rover4 waypoint6)))
(preference o4 (at-most-once (at rover3 waypoint11)))
(preference o5 (at-most-once (at rover3 waypoint0)))
(preference o6 (at-most-once (at rover3 waypoint1)))
(preference o7 (at-most-once (at rover3 waypoint9)))
(preference o8 (at-most-once (at rover2 waypoint9)))
(preference o9 (at-most-once (at rover1 waypoint9)))
(preference o10 (at-most-once (at rover1 waypoint4)))
(preference o11 (at-most-once (at rover0 waypoint2)))
(preference o12 (at-most-once (at rover0 waypoint0)))
(preference o13 (at-most-once (at rover0 waypoint7)))
(preference o14 (at-most-once (empty rover2store)))
(preference o15 (at-most-once (full rover2store)))
(preference o16 (at-most-once (empty rover0store)))
(preference o17 (at-most-once (full rover0store)))
(preference o18 (at-most-once (full rover5store)))
(preference o19 (at-most-once (full rover3store)))
(preference o20 (at-most-once (empty rover1store)))
(preference o21 (at-most-once (full rover1store)))
(preference o22 (at-most-once (calibrated camera6 rover5)))
(preference o23 (at-most-once (calibrated camera3 rover3)))
(preference o24 (at-most-once (calibrated camera1 rover2)))
(preference sb0 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover5 waypoint12)))
(preference sb1 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover5 waypoint1)))
(preference sb2 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover4 waypoint6)))
```

```
(preference sb3 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover3 waypoint11)))
(preference sb4 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover3 waypoint4)))
(preference sb5 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover3 waypoint0)))
(preference sb6 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover2 waypoint9)))
(preference sb7 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover1 waypoint19)))
(preference sb8 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover1 waypoint4)))
(preference sb9 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover1 waypoint7)))
(preference sb10 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover0 waypoint19)))
(preference sb11 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover0 waypoint6)))
(preference sb12 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover0 waypoint14)))
(preference sb13 (sometime-before (have_soil_analysis rover2 waypoint0)
  (at rover0 waypoint0)))
(preference sb14 (sometime-before (have_soil_analysis rover2 waypoint0)
  (full rover2store)))
(preference sb15 (sometime-before (have_soil_analysis rover2 waypoint0)
  (full rover0store)))
(preference sb16 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_soil_analysis rover0 waypoint14)))
(preference sb17 (sometime-before (have_soil_analysis rover2 waypoint0)
  (full rover3store)))
(preference sb18 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_rock_analysis rover3 waypoint4)))
(preference sb19 (sometime-before (have_soil_analysis rover2 waypoint0)
  (full rover1store)))
(preference sb20 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_rock_analysis rover1 waypoint7)))
(preference sb21 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_rock_analysis rover1 waypoint4)))
(preference sb22 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_rock_analysis rover0 waypoint6)))
(preference sb23 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_rock_analysis rover0 waypoint2)))
(preference sb24 (sometime-before (have_soil_analysis rover2 waypoint0)
  (calibrated camera6 rover5)))
(preference sb25 (sometime-before (have_soil_analysis rover2 waypoint0)
  (calibrated camera2 rover4)))
(preference sb26 (sometime-before (have_soil_analysis rover2 waypoint0)
  (calibrated camera1 rover2)))
(preference sb27 (sometime-before (have_soil_analysis rover2 waypoint0)
  (calibrated camera0 rover1)))
(preference sb28 (sometime-before (have_soil_analysis rover2 waypoint0)
  (calibrated camera5 rover0)))
(preference sb29 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_image rover5 objective4 high_res)))
```

```
(preference sb30 (sometime-before (have_soil_analysis rover2 waypoint0)
  (have_image rover2 objective5 colour)))
(preference sb31 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover5 waypoint12)))
(preference sb32 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover4 waypoint6)))
(preference sb33 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover3 waypoint11)))
(preference sb34 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover3 waypoint4)))
(preference sb35 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover3 waypoint0)))
(preference sb36 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover1 waypoint11)))
(preference sb37 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover1 waypoint4)))
(preference sb38 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover1 waypoint7)))
(preference sb39 (sometime-before (have_rock_analysis rover0 waypoint2)
  (at rover0 waypoint14)))
(preference sb40 (sometime-before (have_rock_analysis rover0 waypoint2)
  (full rover2store)))
(preference sb41 (sometime-before (have_rock_analysis rover0 waypoint2)
  (have_soil_analysis rover2 waypoint0)))
(preference sb42 (sometime-before (have_rock_analysis rover0 waypoint2)
  (full rover0store)))
(preference sb43 (sometime-before (have_rock_analysis rover0 waypoint2)
  (have_soil_analysis rover0 waypoint14)))
(preference sb44 (sometime-before (have_rock_analysis rover0 waypoint2)
  (full rover3store)))
(preference sb45 (sometime-before (have_rock_analysis rover0 waypoint2)
  (have_rock_analysis rover3 waypoint4)))
(preference sb46 (sometime-before (have_rock_analysis rover0 waypoint2)
  (full rover1store)))
(preference sb47 (sometime-before (have_rock_analysis rover0 waypoint2)
  (have_rock_analysis rover1 waypoint7)))
(preference sb48 (sometime-before (have_rock_analysis rover0 waypoint2)
  (have_rock_analysis rover1 waypoint4)))
(preference sb49 (sometime-before (have_rock_analysis rover0 waypoint2)
  (calibrated camera6 rover5)))
(preference sb50 (sometime-before (have_rock_analysis rover0 waypoint2)
  (calibrated camera2 rover4)))
(preference sb51 (sometime-before (have_rock_analysis rover0 waypoint2)
  (calibrated camera3 rover3)))
(preference sb52 (sometime-before (have_rock_analysis rover0 waypoint2)
  (calibrated camera1 rover2)))
(preference sb53 (sometime-before (have_rock_analysis rover0 waypoint2)
  (calibrated camera0 rover1)))
(preference sb54 (sometime-before (have_rock_analysis rover0 waypoint2)
  (calibrated camera5 rover0)))
(preference sb55 (sometime-before (have_rock_analysis rover0 waypoint2)
  (have_image rover5 objective3 low_res)))
(preference sb56 (sometime-before (have_rock_analysis rover0 waypoint2)
  (have_image rover5 objective4 high_res)))
```

```
    (preference sb57 (sometime-before (have_rock_analysis rover0 waypoint2)
      (have_image rover3 objective2 colour)))
    (preference sb58 (sometime-before (have_rock_analysis rover0 waypoint2)
      (have_image rover3 objective5 colour)))
    (preference sb59 (sometime-before (have_rock_analysis rover0 waypoint2)
      (have_image rover2 objective3 low_res)))
    (preference sb60 (sometime-before (have_rock_analysis rover0 waypoint2)
      (have_image rover1 objective4 high_res)))
    (preference sb61 (sometime-before (have_rock_analysis rover0 waypoint2)
      (have_image rover0 objective2 colour)))
    (preference sb62 (sometime-before (have_rock_analysis rover0 waypoint2)
      (have_image rover0 objective5 colour)))
  )
)

(:metric minimize
  (+ (* (is-violated sb62) 99) (* (is-violated sb61) 99)
  (* (is-violated sb60) 99) (* (is-violated sb59) 99)
  (* (is-violated sb58) 99) (* (is-violated sb57) 99)
  (* (is-violated sb56) 99) (* (is-violated sb55) 99)
  (* (is-violated sb54) 98) (* (is-violated sb53) 95)
  (* (is-violated sb52) 95) (* (is-violated sb51) 99)
  (* (is-violated sb50) 95) (* (is-violated sb49) 95)
  (* (is-violated sb48) 99) (* (is-violated sb47) 92)
  (* (is-violated sb46) 92) (* (is-violated sb45) 92)
  (* (is-violated sb44) 92) (* (is-violated sb43) 92)
  (* (is-violated sb42) 92) (* (is-violated sb41) 95)
  (* (is-violated sb40) 95) (* (is-violated sb39) 92)
  (* (is-violated sb38) 92) (* (is-violated sb37) 99)
  (* (is-violated sb36) 99) (* (is-violated sb35) 95)
  (* (is-violated sb34) 92) (* (is-violated sb33) 98)
  (* (is-violated sb32) 102) (* (is-violated sb31) 105)
  (* (is-violated sb30) 92) (* (is-violated sb29) 99)
  (* (is-violated sb28) 98) (* (is-violated sb27) 95)
  (* (is-violated sb26) 92) (* (is-violated sb25) 95)
  (* (is-violated sb24) 92) (* (is-violated sb23) 92)
  (* (is-violated sb22) 92) (* (is-violated sb21) 99)
  (* (is-violated sb20) 92) (* (is-violated sb19) 92)
  (* (is-violated sb18) 92) (* (is-violated sb17) 92)
  (* (is-violated sb16) 92) (* (is-violated sb15) 92)
  (* (is-violated sb14) 92) (* (is-violated sb13) 104)
  (* (is-violated sb12) 92) (* (is-violated sb11) 92)
  (* (is-violated sb10) 92) (* (is-violated sb9) 92)
  (* (is-violated sb8) 99) (* (is-violated sb7) 92)
  (* (is-violated sb6) 104) (* (is-violated sb5) 95)
  (* (is-violated sb4) 92) (* (is-violated sb3) 92)
  (* (is-violated sb2) 102) (* (is-violated sb1) 104)
  (* (is-violated sb0) 105) (* (is-violated o24) 99)
  (* (is-violated o23) 104) (* (is-violated o22) 92)
  (* (is-violated o21) 92) (* (is-violated o20) 92)
  (* (is-violated o19) 92) (* (is-violated o18) 92)
  (* (is-violated o17) 98) (* (is-violated o16) 98)
  (* (is-violated o15) 92) (* (is-violated o14) 92)
  (* (is-violated o13) 99) (* (is-violated o12) 95)
```

```
    (* (is-violated o11) 98) (* (is-violated o10) 99)
    (* (is-violated o9) 98) (* (is-violated o8) 98)
    (* (is-violated o7) 95) (* (is-violated o6) 105)
    (* (is-violated o5) 99) (* (is-violated o4) 92)
    (* (is-violated o3) 95) (* (is-violated o2) 98)
    (* (is-violated o1) 102) (* (is-violated o0) 98)
    (* (is-violated e53) 95) (* (is-violated e52) 95)
    (* (is-violated e51) 95) (* (is-violated e50) 92)
    (* (is-violated e49) 92) (* (is-violated e48) 92)
    (* (is-violated e47) 95) (* (is-violated e46) 95)
    (* (is-violated e45) 95) (* (is-violated e44) 95)
    (* (is-violated e43) 95) (* (is-violated e42) 92)
    (* (is-violated e41) 95) (* (is-violated e40) 98)
    (* (is-violated e39) 95) (* (is-violated e38) 95)
    (* (is-violated e37) 95) (* (is-violated e36) 92)
    (* (is-violated e35) 95) (* (is-violated e34) 95)
    (* (is-violated e33) 98) (* (is-violated e32) 92)
    (* (is-violated e31) 92) (* (is-violated e30) 102)
    (* (is-violated e29) 104) (* (is-violated e28) 92)
    (* (is-violated e27) 95) (* (is-violated e26) 92)
    (* (is-violated e25) 95) (* (is-violated e24) 102)
    (* (is-violated e23) 102) (* (is-violated e22) 95)
    (* (is-violated e21) 92) (* (is-violated e20) 92)
    (* (is-violated e19) 92) (* (is-violated e18) 95)
    (* (is-violated e17) 92) (* (is-violated e16) 95)
    (* (is-violated e15) 95) (* (is-violated e14) 92)
    (* (is-violated e13) 102) (* (is-violated e12) 95)
    (* (is-violated e11) 99) (* (is-violated e10) 98)
    (* (is-violated e9) 104) (* (is-violated e8) 102)
    (* (is-violated e7) 95) (* (is-violated e6) 92)
    (* (is-violated e5) 95) (* (is-violated e4) 98)
    (* (is-violated e3) 95) (* (is-violated e2) 92)
    (* (is-violated e1) 92) (* (is-violated e0) 99)
    (* (is-violated a5) 95) (* (is-violated a4) 104)
    (* (is-violated a3) 92) (* (is-violated a2) 92)
    (* (is-violated a1) 99) (* (is-violated a0) 92)
    )
  )
)
```

Planification avec préférences basée sur la Théorie de l'Utilité Multi-Attribut couplée à une intégrale de Choquet : application à l'interopérabilité des organisations en gestion de crise

Résumé : Cette étude s'intéresse à la prise en compte des préférences des décideurs dans le cadre de la résolution des problèmes de planification. L'originalité de l'approche retenue consiste à représenter les préférences en utilisant un formalisme issu de l'aide à la décision multicritère à savoir un modèle MAUT (acronyme de Théorie de l'Utilité Multi-Attribut) couplé à une intégrale de Choquet. Ce formalisme généralise la notion de préférence utilisée en PDDL (le Langage de Définition des Domaines de Planification). Ainsi, l'extension PDDL3/MAUT proposée enrichit le pouvoir expressif du PDDL en permettant d'utiliser facilement un nombre quelconque de préférences numériques, d'agréger des préférences entre elles ou encore de considérer d'éventuelles interactions entre les préférences du problème. En conséquence, elle permet de représenter plus finement la complexité intrinsèque des préférences des décideurs. Par ailleurs, un algorithme pour la résolution des problèmes de planification avec préférences est proposé. Ce dernier est mis en oeuvre dans le planificateur ChoPlan qui a été développé dans le cadre de cette étude et dont les performances ont été comparées à celles des planificateurs de l'art. En outre, les travaux réalisés contribuent à la résolution de la problématique de l'interopérabilité des organisations en situation de crise. En effet, un système d'aide à la décision capable de supporter les décideurs lors de l'élaboration de plans d'action collaboratifs est présenté. Ce dernier permet de modéliser la situation à résoudre, les capacités des partenaires mobilisés ainsi que les objectifs, contraintes et préférences des décideurs. Les modèles ainsi réalisés sont ensuite transformés afin de générer un problème de planification avec préférences qui est résolu à l'aide de ChoPlan.

Mots-clés : Planification avec préférences, Aide à la décision multicritère, Gestion de crise, Plan collaboratif, Processus collaboratif.

Preference-based planning using the Multi-Attribute Utility Theory along with a Choquet integral : application to organizations' interoperability in crisis management

Abstract : This study aims to solve preference-based planning problems. The originality of this work is to represent preferences using a formalism from multicriteria decision analysis namely a MAUT model (acronym for Multi-Attribute Utility Theory) along with a Choquet integral. This formalism generalizes the notion of preference used in PDDL (the Planning Domain Definition Language). Indeed, the proposed PDDL3/MAUT extension improves the PDDL expressiveness by allowing to use any number of numeric preferences, aggregating preferences and considering interactions between preferences. As a consequence, it can represent more accurately the intrinsic complexity of decision-makers preferences. Furthermore, an algorithm for preference-based planning has been designed. It has been used to implement a planner named ChoPlan whose performances have been compared to state of the art planners. In addition, this work adresses the problem of organization's interoperability in crisis management. Indeed, a decision aid system supporting decision-makers during the design of collaborative plans is presented. It helps stakeholders to model the situation to solve, the responders' capabilities as well as objectives, constraints and preferences of the decision-makers. These models are then processed to generate a preference-based planning problem that is solved using the ChoPlan planner.

Keywords : Preference-based planning, Multicriteria decision analysis, Crisis management, Collaborative plan, Collaborative process.