Décisions multicritères dans les réseaux de télécommunications autonomes

Farouk Aissanou

Ecole Doctorale EDITE

Thèse présentée pour l'obtention du diplôme de Docteur de Télécom SudParis

Doctorat conjoint Télécom SudParis et Université Pierre et Marie Curie

Spécialité :

Informatique

Par

Farouk AISSANOU

Titre

Décisions multicritères dans les réseaux de télécommunication autonomes

Soutenue le 19 juin 2012 devant le jury composé de :

Patrick Siarry	Rapporteur	Université Paris-Est Créteil Val-de-Marne
Khaldoun Al Agha	Rapporteur	Université Paris-Sud
Pierre Sens	Examinateur	Université Pierre et Marie Curie
Xavier Gandibleux	Examinateur	Université de Nantes
Alain Petrowski	Encadrant	Institut Mines-Télécom, Télécom SudParis
Djamal Zeghlache	Directeur de thèse	Institut Mines-Télécom, Télécom SudParis

Thèse numéro : 2012TELE0019

Remerciements

D'abord, je tiens à remercier spécialement mon directeur de thèse : Monsieur Djamal Zeghlache de l'Institut Télécom, mon encadrant : Monsieur Alain Petrowski de l'Institut Télécom et Madame Ilham Benyahia de l'Université du Québec en Outaouais qui m'ont guidés et aidés dans mon travail tout au long de cette thèse. Cette thèse n'aurait jamais vu le jour sans leur aide en termes d'idées, de conseils et de discussions fructueuses.

Je tiens à remercier Monsieur Khaldoun Al Agha et Monsieur Patrick Siarry d'avoir rapporté sur la thèse et d'avoir contribué à améliorer la qualité de ce mémoire.

Je tiens à remercier Monsieur Khaldoun Al Agha, Monsieur Patrick Siarry, Monsieur Xavier Gandibleux et Monsieur Pierre Sens d'avoir acceptés de participer au jury de cette thèse.

Je remercie du fond du cœur ma mère et mon père qui ont été présents pendant tout le cursus de mes études. Leur apport est inestimable.

Je tiens aussi à remercier mes frères, mes sœurs et mes proches qui n'ont jamais cessé de m'encourager durant mes études.

Finalement, je tiens à remercier tous mes amis et collègues qui ont contribué de près ou de loin à l'accomplissement de ce travail.

Résumé

Les réseaux de données actuels sont des entités complexes qui opèrent dans des environnements dynamiques et hétérogènes. L'architecture et les protocoles de ces réseaux doivent faire face à plusieurs défis, notamment l'adaptation dynamique et la prise de décisions autonome en présence de plusieurs critères, souvent contradictoires, tels que le délai, le taux de perte, la gigue, l'énergie, etc. Cependant, les problèmes de décision multicritère ont généralement de multiples solutions. Ces problèmes sont résolus par des méthodes qui utilisent des paramètres dont les choix ont des conséquences difficiles à prévoir. De ce fait, la plupart des méthodes de décision multicritère proposées dans la littérature supposent la présence d'un décideur qui guide le processus de décision. Enfin, le choix des paramètres suppose souvent une interactivité avec le décideur, ce qui est difficile, voire impossible, à envisager dans un contexte autonome.

Dans cette thèse, nous proposons une nouvelle méthode de décision multicritère adaptée aux systèmes autonomes en général et aux réseaux autonomes en particulier. La méthode de décision multicritère de type "poupée russe" que nous introduisons utilise un ensemble de boîtes de qualité englobantes, définies dans l'espace des critères, afin d'estimer une large gamme de fonctions d'utilité. D'une part, la méthode proposée s'adapte au caractère dynamique des réseaux autonomes, afin de maximiser la satisfaction des utilisateurs. D'autre part, elle utilise des paramètres qui sont soit directement déduits de faits objectifs, tels que des normes ou spécifications techniques, soit obtenus à l'aide d'une expérience de type MOS (*Mean Opinion Score*) au moyen d'une méthode de classification automatique.

Nous avons testé les performances de la méthode de la poupée russe sur un cas pratique de routage dans les réseaux ad hoc sans fil. Les expérimentations ont montré que le routage réalisé avec la méthode de la poupée russe est toujours meilleur ou similaire à celui de la méthode de la somme pondérée qui est largement utilisée. Cela est dû à la capacité d'adaptation de la décision offerte par cette nouvelle méthode de décision multicritère.

Mots clés :

Méthode de décision multicritère, réseaux autonomes, réseaux adaptatifs, réseaux sans fil, routage multicritère.

Abstract

Today's data networks are complex entities that operate in dynamic and heterogeneous environments. The architecture and protocols of these networks have to face several challenges such as dynamic adaptation and autonomous decision-making in the presence of several, often conflicting, criteria such as delay, loss rate, jitter, energy, etc. However, multicriteria decision making problems usually have multiple solutions. These problems are solved by methods that use parameters whose choices have consequences difficult to predict. Thus, most multicriteria decision making methods proposed in the literature assume the presence of a decision maker who guides the decision process. Finally, the choice of parameter values often involves an interaction with the decision maker, which is difficult or impossible to do in an autonomous context.

In this thesis, we propose a new multicriteria decision making method suitable for autonomous systems in general and autonomous networks in particular. The Russian doll like method we propose uses a set of nested quality boxes (like Russian dolls) defined in the criteria space, in order to approximate a wide range of utility functions. First, the proposed method adapts to the dynamic nature of autonomous networks in order to maximize user satisfaction. Second, it uses parameters that are directly deduced from objective facts such as technical standards or specifications, or obtained from a MOS (Mean Opinion Score) experiment using an automatic classification method.

We tested the performance of the Russian doll like method in a case of routing in wireless ad hoc networks. Experiments have shown that routing done with the Russian doll like method is always better or similar to the one done by the weighted sum method which is widely used. This is due to the adaptation ability of the decision provided by this new multicriteria decision making method.

Keywords :

Multicriteria decision making method, autonomous networks, adaptive networks, wireless networks, multicriteria routing.

Table des matières

Chapitre 1

Introduction

1.1 Contexte et problématique

Les réseaux actuels sont des systèmes complexes constitués d'un grand nombre de composants hétérogènes et une communauté d'utilisateurs large et variée. Ces systèmes sont le résultat de l'adaptation constante des protocoles de l'architecture protocolaire classique (OSI et TCP/IP) aux nouveaux défis des réseaux de données, en l'occurrence : la convergence voix, données et vidéo, la mobilité et le fonctionnement dans un environnement hostile, tel qu'un environnement sans fil. Les changements apportés à l'architecture en couches du réseau sont spécifiques aux caractéristiques des applications visées, nous pouvons citer à titre d'exemple la prise en charge de la mobilité dans la couche réseau du protocole TCP/IP (Protocole IP mobile) [Per02] ou l'introduction de protocoles spécifiques aux applications multimédias (RTP/RTCP) [Sch03] dans la couche transport du standard TCP/IP. Ces nouveaux protocoles sont souvent conçus en respectant le principe de base des protocoles en couches, à savoir la hiérarchisation et l'abstraction des services échangés par les couches uniquement adjacentes.

Bien que ce principe de limitation de la communication entre couches adjacentes, ait permis un développement considérable des réseaux câblés, il constitue, néanmoins, un véritable obstacle face à l'émergence des réseaux sans fil. En effet, les caractéristiques d'un réseau sans fil sont complètement différentes de celles d'un réseau filaire. Par exemple, le taux d'erreur binaire est relativement élevé dans un réseau sans fil, contrairement, à celui du réseau filaire qui reste acceptable. Par conséquent, la ma-

jorité des protocoles qui ont été conçus pour un réseau câblé ne peuvent fonctionner normalement dans un environnement sans fil. Le protocole de contrôle de congestion dans un réseau IP constitue un exemple illustrant cette problématique. En effet, le fonctionnement de ce protocole est fondé sur le principe qui stipule qu'un paquet perdu dans un réseau filaire induit l'apparition d'une congestion dans le réseau, ce qui entraîne systématiquement une diminution du débit de transmission. Dans un réseau sans fil, les pertes de paquets sont souvent dues aux transmissions erronées, dans ce cas il faut retransmettre rapidement les paquets erronés (i.e. augmenter le débit) ce qui est tout à fait contradictoire avec le principe de fonctionnement du protocole de contrôle de congestion. Ce type de problème, associé à ceux relatifs à la mobilité et à la limitation de l'énergie dans un réseau sans fil, a soulevé une question importante, qui est celle de l'utilité du modèle en couches classique pour un réseau sans fil. Les recherches récentes montrent que la communauté scientifique se dirige de plus en plus vers une rupture définitive avec l'ancienne architecture, en s'intéressant de près à de nouveaux types de réseaux, en l'occurrence les réseaux autonomes ou cognitifs.

Les réseaux autonomes/cognitifs [Mah07] sont des réseaux capables de percevoir les conditions de leur environnement afin de planifier, puis prendre des décisions et enfin agir pour s'y adapter [TFDM06]. Les réseaux cognitifs sont fondés sur le concept de cycle cognitif introduit par J. Mitola [Mit00]. La figure 1.1, montre le concept de cycle cognitif qui a été largement repris dans la littérature. Ce dernier est défini par cinq phases successives et un élément central qui est l'apprentissage. Nous remarquons que le cycle cognitif est assez flexible, du fait qu'il permet une adaptation rapide par rapport à la nature de l'événement reçu de l'environnement. Le réseau en tant qu'entité de l'environnement communique avec son environnement externe en exécutant des actions et en recevant des événements qui traduisent l'état courant de l'environnement, ainsi que celui du réseau. Cet échange permanent entre le réseau et l'environnement externe est traduit par le cycle cognitif du réseau, qui n'est autre qu'une boucle de rétroaction semblable à celles présentes dans la plupart des systèmes autonomes.

Jusqu'à présent les chercheurs se sont focalisés sur la conception des systèmes autonomes/cognitifs en termes d'architecture et d'apprentissage et raisonnement. En effet,

il y a peu de travaux qui se sont intéressés à la prise de décision en général et la prise de décision multicritère en particulier. Cependant, la majorité des fonctionnalités assurées par un système autonome sont de nature multicritère, à cause de l'aspect contradictoire des métriques associées [MRA+09]. Par exemple une fonctionnalité de transfert de fichier exige un taux de perte réduit ; par contre elle est nettement tolérante en termes de délai de transfert de bout en bout. D'un autre coté, une application de téléphonie sur internet peut être tolérante en terme de taux de perte, mais extrêmement exigeante en terme de délai. De plus, la nature de transmission dans un réseau sans fil entraîne une contradiction des métriques. En effet, puisque le taux de perte est directement lié à la distance, un terminal sans fil aura tendance à relier un paquet donné à un voisin direct au lieu de le transmettre directement à une destination éloignée, ce qui entraîne une augmentation du délai de transfert de bout en bout et une grande consommation d'énergie le long du chemin de communication. Ainsi, les réseaux autonomes/cognitifs doivent être capables de gérer les problématiques relatives aux décisions multicritères.

Les méthodes de décision multicritères classiques exigent la présence d'un décideur, qui est généralement un opérateur ou un groupe d'opérateurs humains. La plupart de ces méthodes utilisent des poids de quantification de l'importance des différents critères/objectifs dans le processus de décision multicritère. Cependant, il est difficile, voire impossible de trouver une correspondance optimale entre la valeur des poids de quantification et l'importance des critères/objectifs dans le processus de décision multicritère. De plus, l'environnement dynamique dans lequel opèrent les réseaux autonomes/cognitifs entraîne une modification dynamique de l'importance de chaque critère au cours du temps. En effet, l'importance d'un critère donné peut augmenter considérablement dès que le réseau ou le système autonome entre dans un état de fonctionnement critique [MA10]. Par conséquent, même si la valeur initiale des poids de quantification est optimale, elle ne le restera pas, sans doute, au cours du temps. Enfin, l'aspect autonome implique l'absence d'un opérateur humain ; par conséquent, toutes les méthodes qui sont fondées sur l'interaction avec un décideur ne peuvent pas être utilisées dans un contexte de réseaux autonomes/cognitifs.

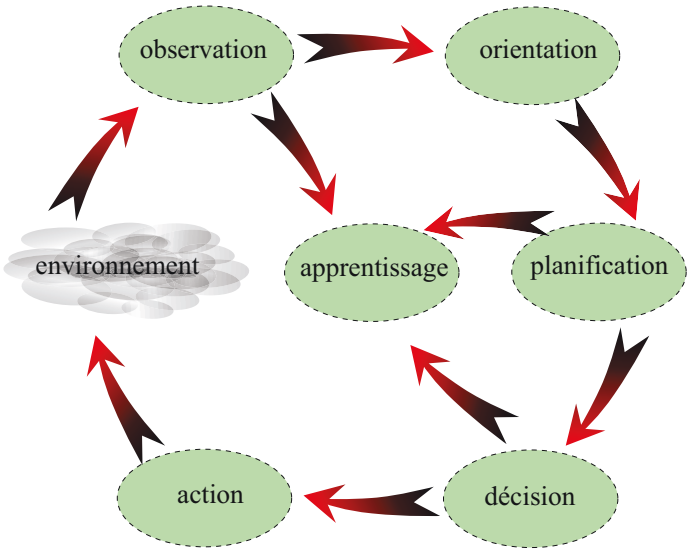


FIGURE 1.1 – Cycle cognitif

1.2 Motivation

Notre objectif dans cette thèse est de concevoir une méthode de décision multicritère capable de gérer le processus de décision multicritère dans un contexte de systèmes autonomes en général, et de réseaux autonomes en particulier. La méthode de décision multicritère doit respecter les exigences suivantes :

- elle doit permettre au système de prendre, souvent, des décisions multicritères de bonne qualité qui lui assurent un bon fonctionnement global ;
- elle doit être capable de fonctionner indépendamment d'un décideur. En d'autres termes, les paramètres de la méthode doivent être fixés par le concepteur du système au moment de l'initialisation de ce dernier, sans pour autant exiger une interaction avec un décideur au cours du temps ;
- elle doit utiliser des paramètres faciles à interpréter et faciles à déterminer. Aussi, ces paramètres doivent être reconfigurables d'une façon automatique en fonction des données remontées par le mécanisme d'observation du cycle cognitif ;
- elle doit être adaptative au contexte dynamique de fonctionnement d'un réseau autonome.

1.3 Contributions

Les contributions majeures de cette thèse sont les suivantes :

- Nous proposons une nouvelle méthode de décision multicritère, appelée méthode de la poupée russe, qui est utilisée dans un contexte de réseaux autonomes. La méthode de la poupée russe utilise un ensemble de boîtes de qualité englobantes afin de modéliser une large classe de fonctions d'utilité. La méthode de la poupée russe tient son avantage de sa facilité d'utilisation, vu que ses paramètres sont directement déduits soit des normes ou des documents techniques soit d'une expérience de type sondage d'opinions MOS (*Mean Opinion Score*) au moyen d'une méthode de classification automatique.
- Nous appliquons la méthode de la poupée russe pour traiter la problématique de routage multicritère dans les réseaux ad hoc sans fil.

1.4 Plan de la thèse

La suite de cette thèse est présentée comme suit :

- Chapitre 2 : dans ce chapitre nous donnons un état de l'art des méthodes de décision multicritère classiques et nous les discutons relativement à leur application dans un contexte de réseaux autonomes. En outre, nous introduisons un ensemble de définitions relatives au domaine de l'optimisation multiobjectif et de la décision multicritère.
- Chapitre 3 : ce chapitre introduit la méthode de la poupée russe. Il est organisé comme suit : d'abord, nous donnons une vue générale de la méthode, puis nous exposons formellement cette dernière. Enfin, nous introduisons les différentes variantes de la méthode, selon la relation entre les critères impliqués dans le processus de décision multicritère.
- Chapitre 4 : dans ce chapitre, nous appliquons la méthode de la poupée russe au cas du routage dans les réseaux ad hoc sans fil. La méthode de la poupée russe est utilisée, dans ce contexte, conjointement avec un protocole de routage basé sur l'apprentissage par renforcement, afin de trouver des chemins multicritères selon

les exigences des applications routées dans le réseau. En outre, nous présentons des simulations montrant les performances de la méthode de la poupée russe en comparaison de la méthode de la somme pondérée.

– Chapitre 5 : dans ce chapitre, nous présentons des conclusions et des perspectives pour le travail à venir.

Chapitre 2

État de l'art des méthodes de décision multicritère

2.1 Introduction

La problématique de décision multicritère est souvent présente dans la vie pratique. Du simple choix d'un achat à la sélection d'une carrière, la question demeure la même : comment faire le bon choix en tenant compte de toutes les contradictions qui existent dans les critères qui participent au processus de décision ? La problématique de décision multicritère se réfère à une prise de décision en présence de plusieurs critères, souvent contradictoires. Par exemple, l'achat d'une nouvelle voiture consiste à trouver le ou les meilleurs modèles qui offrent un bon compromis entre le prix, la consommation de carburant, le confort, etc. On remarquera ici que les critères de confort et de niveau de consommation sont souvent contradictoires : une voiture confortable est supposée être spacieuse, par conséquent lourde, ce qui obligera le constructeur à incorporer un moteur puissant adapté aux dimensions de la voiture. On remarquera aussi que les natures de ces deux critères sont totalement différentes : d'un côté le confort est un critère qualitatif alors que le niveau de consommation est quantitatif. De plus, les critères quantitatifs peuvent être incommensurables, c'est-à-dire qu'il n'existe pas de justification permettant de les agréger en un seul critère. La puissance et le prix d'une voiture donnée sont des exemples de deux critères incommensurables, parce que

ramener une puissance à un coût ne répond pas au besoin de l'utilisateur. Ceci dit, la problématique de décision multicritère est d'autant plus complexe qu'elle a engendré de multiples travaux de recherches, qui ont donné naissance à de multiples méthodes de décision multicritère.

Les systèmes autonomes en général et les réseaux autonomes en particulier sont des systèmes complexes qui visent à se distancier d'un opérateur humain pour leur fonctionnement. Dans un contexte de décision multicritère, un système autonome doit prendre des décisions dans des contextes variés, sans l'aide directe d'un décideur. De plus, les réseaux autonomes fonctionnent dans un environnement dynamique, dans lequel des décisions doivent être prises en temps réel. Cet ensemble de contraintes doit être pris en compte par les méthodes de décision multicritère, afin qu'elles puissent être utilisables dans un contexte autonome.

Dans ce chapitre, nous exposerons les principes fondamentaux de l'optimisation multiobjectif, ainsi que de la décision multicritère. Nous illustrerons en détail le fonctionnement des principales méthodes.

2.2 Définitions

2.2.1 Alternative

Une alternative ou action désigne un objet sur lequel opérera le processus de décision.

2.2.2 Critère

Un critère, objectif ou attribut est une fonction définie sur l'ensemble des alternatives, qui prend ses valeurs dans un ensemble totalement ordonné et qui représente les préférences de l'utilisateur selon le point de vue que le critère modélise [RV89].

2.2.3 Décision multicritère

Le domaine de la décision multicritère MCDM (*Multicriteria Decision Making*) se décompose en deux sous-domaines, en l'occurrence la décision multiobjectif MODM (*Multi-Objective Decision Making*) et la décision multi-attribut MADM (*Multi-Attribute Decision Making*) [Kla09] :

- MADM : sélection de la meilleure alternative dans un ensemble fini prédéterminé d'alternatives ;
- MODM : sélection de la meilleure action dans un espace de décisions continu ou discret [Kla09]. L'optimisation multiobjectif est une branche de MODM.

2.2.4 Problème d'optimisation multiobjectif

Comme nous l'avons introduit ci-dessus, le problème d'optimisation multiobjectif consiste à sélectionner une ou plusieurs solutions qui présentent un bon compromis entre plusieurs critères contradictoires. Mathématiquement parlant, le problème d'optimisation multiobjectif se définit comme suit, dans le cas d'une minimisation :

$$\begin{cases} \text{Minimiser} : \mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})]^T \\ \text{s.t. } g_j(\mathbf{x}) \leq 0; j = 1, 2, ..., m \end{cases}$$

où k est le nombre de fonctions objectifs et m est le nombre de contraintes. $\mathbf{x} \in E^n$ est un vecteur de variables de décision, et $\mathbf{F}(\mathbf{x}) \in E^k$ est un vecteur de fonctions objectifs $f_i(\mathbf{x}) : E^n \to E^1$. $\mathbf{X} = \{\mathbf{x}, g_j(\mathbf{x}) \leq 0, j = 1, ..., m\}$ est l'espace réalisable des actions, et $\mathbf{C} = \{\mathbf{F}(\mathbf{x}), \mathbf{x} \in \mathbf{X}\}$ est l'espace réalisable des critères.

Contrairement aux problèmes d'optimisation mono-objectif, la solution à un problème multiobjectif est souvent un ensemble de solutions dites solutions non-dominées. Ces solutions sont meilleures que les solutions dominées, mais elles sont incomparables mutuellement selon la relation de dominance. L'ensemble des solutions non-dominées est défini par une relation de dominance. La dominance de Pareto est fréquemment utilisée dans le domaine de l'optimisation multiobjectif, elle est décrite dans la sous-section 2.2.9.

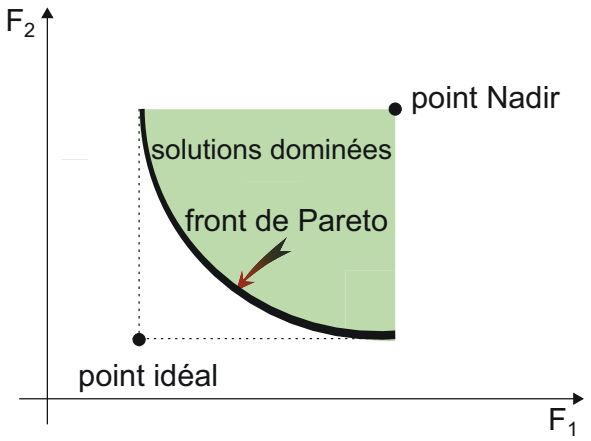


FIGURE 2.1 – Éléments de l'optimisation multiobjectif (cas de deux objectifs)

2.2.5 Point idéal

Le point idéal (voir figure 2.1) est un vecteur dont les coordonnées sont les minimums de tous les critères. Il est clair que le point idéal n'est généralement pas atteignable quand les critères sont contradictoires. Le point idéal peut être utilisé comme point de référence que l'on cherche à atteindre.

2.2.6 Point Nadir

Les coordonnées du point Nadir correspondent, dans le cas d'une minimisation de tous les critères, aux maxima des critères (voir figure 2.1). Ce point est utile pour normaliser les critères.

2.2.7 Décideur

Le décideur est généralement une personne ou un groupe de personnes qui sont supposés connaître le problème de décision multicritère. Le décideur se base sur son expérience et ses connaissances pour exprimer des relations de préférence entre différentes solutions. Dans ce contexte, le décideur est souvent épaulé par un analyste, qui joue le rôle d'interface entre le décideur et l'aspect mathématique du processus de décision multicritère.

2.2.8 Fonction d'utilité

Il est souvent supposé qu'un décideur se base sur une certaine fonction pour formuler ses préférences [Mie99]. Cette fonction est appelée fonction d'utilité ou fonction de valeur. Une fonction d'utilité est une fonction $u : \mathbf{R}^k \mapsto \mathbf{R}$ qui traduit les préférences du décideur comme suit.

Soit $\mathbf{z}_1$ et $\mathbf{z}_2$ deux vecteurs de critères : $\mathbf{z}_1$ est meilleur que $\mathbf{z}_2$ si $u(\mathbf{z}_1) > u(\mathbf{z}_2)$, $\mathbf{z}_1$ et $\mathbf{z}_2$ sont indifférents si $u(\mathbf{z}_1) = u(\mathbf{z}_2)$.

Il est à noter que si l'on connaît l'expression mathématique exacte de la fonction d'utilité, nous pouvons traiter le problème décrit dans la section 2.2.4 comme un problème de décision monocritère. Ainsi, nous pouvons appliquer les méthodes d'optimisation classiques pour le résoudre. Cependant, il est *a priori* difficile, voire impossible, de cerner le processus de décision d'un décideur avec une fonction mathématique.

2.2.9 Dominance de Pareto

La dominance de Pareto est définie dans l'espace des critères $\mathbf{C}$. Soit $\mathbf{a} = (a_1, ..., a_k)$ et $\mathbf{b} = (b_1, ..., b_k)$ deux vecteurs de critères définis dans l'espace des critères $\mathbf{C}$. $\mathbf{a}$ domine $\mathbf{b}$ (dans le cas d'une minimisation) si et seulement si :

$$a_i \leq b_i, \forall i \in [1, k] \text{ et } \exists j \in [1, k], a_j < b_j$$

Soit $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), ..., f_k(\mathbf{x}))$ les fonctions objectifs d'un problème d'optimisation multiobjectif. La solution à ce problème consiste à trouver l'ensemble des vecteurs $\mathbf{X}^*$ tel que chaque vecteur de critères $\mathbf{z} = \mathbf{F}(\mathbf{x})$ avec $\mathbf{x} \in \mathbf{X}^*$ est non dominé. Le front de Pareto $\mathcal{F}$ est l'ensemble $\mathcal{F} = \{\mathbf{F}(\mathbf{x}), \mathbf{x} \in \mathbf{X}^*\}$. La figure 2.1 illustre un front de Pareto dans un espace de critères à deux dimensions.

La problématique de décision multicritère dans ce cas consiste à choisir un point du front de Pareto qui s'apparente le mieux avec les préférences du décideur. Par conséquent, la formalisation des préférences du décideur est un élément crucial du processus de décision multicritère. Cette formalisation est prise en compte par les

différentes méthodes de décision multicritère que nous présentons dans les sections qui suivent.

2.3 Classification des méthodes de décision multicritère

Les méthodes de décision multicritère introduites dans la littérature se distinguent les unes par rapport aux autres selon le moment où les préférences du décideur sont utilisées. Dans ce contexte, on distingue trois types de méthodes, à savoir les méthodes à préférence *a priori*, les méthodes à préférence progressive et enfin les méthodes à préférence *a posteriori* [Hor97].

- Les méthodes à préférence *a priori* : dans ce type de méthodes, les préférences du décideur sont spécifiées avant le lancement du processus de décision multicritère. A la fin de l'optimisation, la méthode est supposée fournir une décision qui traduit le mieux les préférences de l'utilisateur. Les méthodes scalaires sont des instances importantes de cette famille. Ces méthodes sont présentées en détail dans la section 2.4.
- Les méthodes à préférence progressive font intervenir le décideur tout au long de l'exécution du processus de décision. En effet, dans ce type de méthodes, le décideur est amené à spécifier ses préférences en fonction de l'évolution du processus de décision. Les méthodes interactives sont des instances de cette famille.
- Les méthodes à préférence *a posteriori* font un travail au préalable, en trouvant les solutions non-dominées qui s'approchent au mieux du front de Pareto. Ensuite, le choix final est laissé au décideur, qui devra sélectionner la meilleure solution non-dominée qui correspond le mieux à ses préférences. Les méthodes évolutionnaires sont des exemples de cette famille (voir section 2.6).

Dans les sections qui suivent, nous illustrons le fonctionnement de quelques méthodes de décision multicritère qui sont largement abordées dans la littérature. Pour une illustration exhaustive, nous invitons le lecteur à consulter les ouvrages [Mie99, CS02, Ehr05, BDM08]. Enfin et afin de ne pas alourdir la présentation, nous considérons que le décideur désire minimiser tous les objectifs/critères du problème traité.

2.4 Méthodes scalaires

2.4.1 Méthode de la somme pondérée

La méthode de la somme pondérée est à la fois la méthode la plus simple et la plus connue dans le domaine de prise de décision multicritère. Comme son nom l'indique, la méthode de la somme pondérée consiste à pondérer les différents critères du problème de décision multicritère avec des nombres réels appelés poids, qui représentent l'importance de chaque critère dans le processus de décision. Une fois l'importance des différents critères quantifiée, la méthode choisit l'action qui minimise ou maximise la somme pondérée des critères. Mathématiquement parlant, la méthode de la somme pondérée consiste à minimiser une fonction u définie comme suit :

$$u(\mathbf{x}) = \sum_{i=1}^{k} w_i f_i(\mathbf{x})$$

Graphiquement, la solution optimale donnée par la méthode de la somme pondérée est le point tangent d'un hyperplan d'équation $\sum_{i=1}^{k} w_i f_i(\mathbf{x}) = c$ avec le front de Pareto du problème considéré, tel que c est minimum et $\mathbf{x}$ est une action. La figure 2.2, illustre la solution donnée par la méthode de la somme pondérée dans un espace de critères à deux dimensions. La solution donnée par la méthode de la somme pondérée est Pareto optimale si tous les poids utilisés sont strictement positifs [Zad63]. Selon les valeurs des poids w_i, la méthode peut trouver toutes les solutions appartenant au front de Pareto si l'espace réalisable des critères est convexe [AP96]. En effet, comme le montre la figure 2.3, le point $\mathbf{c}_1$ qui est situé sur la partie non-convexe du front de Pareto ne minimise pas la fonction u. Cette situation est problématique quand les points appartenant à

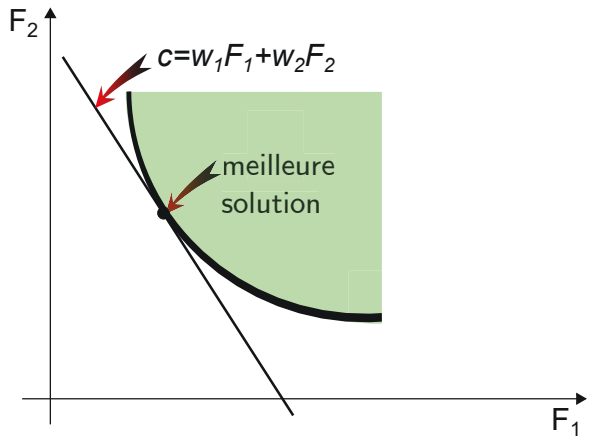


FIGURE 2.2 – Représentation graphique de la méthode de la somme pondérée

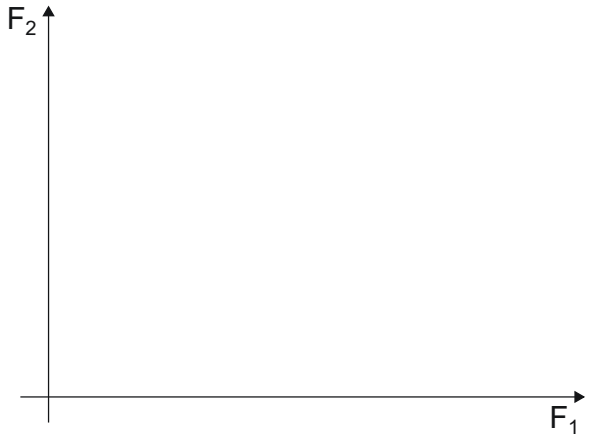


FIGURE 2.3 – $\mathbf{c}_1$ est une solution non-dominée non-optimale pour une fonction d'utilité linéaire

la partie non-convexe du front de Pareto sont les solutions qui s'apparentent le mieux aux préférences du décideur.

Bien que la méthode de la somme pondérée soit assez simple et facile à utiliser, elle souffre, néanmoins, d'une difficulté relative au choix des valeurs des poids de pondération. En effet, puisque les poids sont supposés traduire les préférences de l'utilisateur, les performances de toute la méthode en dépendent.

Un problème important de la somme pondérée est la nature linéaire de l'approximation de la fonction d'utilité. En effet, la modélisation des préférences du décideur avec une fonction linéaire peut conduire à des décisions éloignées de ses préférences, en raison d'une modélisation trop grossière. De plus, modéliser les préférences du dé-

cideur par une fonction linéaire suppose que l'importance de chaque critère vis-à-vis des autres est constante, quelles que soient les valeurs prises par les critères. Cependant, cela est rarement le cas dans la pratique. Par exemple, dans le cas de la prise de décision dans des systèmes complexes, l'importance d'un critère donné peut changer en fonction de l'état du système (fonctionnement critique ou normal) [MA10].

Choix des valeurs des poids

Comme nous l'avons introduit ci-dessus, il est souvent difficile de faire une correspondance directe entre les préférences du décideur et les valeurs des poids de pondération des critères. Néanmoins, des chercheurs ont développé plusieurs méthodes qui ont pour but d'aider le décideur à choisir la valeur des poids. Dans ce contexte, il existe deux classes d'approches, à savoir les approches fondées sur le principe de classement (*ranking methods*) [YH95] et les approches fondées sur le principe de comparaison deux à deux des alternatives (*paired comparison methods*) [SH98].

Dans les approches de classement, le décideur est amené à ordonner les critères selon leur importance, du moins important au plus important. Ensuite, l'approche affecte un poids égal à 1 pour le critère le moins important et des entiers incrémentés successivement pour le reste des poids pondérant les autres critères.

Dans les approches fondées sur le principe de comparaison deux à deux, le décideur est amené à comparer les alternatives deux à deux pour constituer une matrice de comparaison. Dans [SH98], les auteurs proposent d'utiliser le vecteur propre (normalisé à 1) correspondant à la valeur propre maximale de la matrice de comparaison deux à deux comme valeur des poids de pondération des différents critères du problème.

Enfin, dans [RR89], les auteurs introduisent une nouvelle méthode de fixation des poids au moyen de la théorie des ensembles flous.

2.4.2 Théorie de l'utilité multi-attribut

La théorie de l'utilité (M.A.U.T, *Multi Attribute Utility Theory*) a fait l'objet de nombreux travaux, dont ceux de Fishburn [Fis70] et de Keeney-Raiffa [KR76]. Les méthodes issues de la théorie de l'utilité sont fondées sur le principe de la fonction

d'utilité. Les méthodes MAUT tentent de modéliser les préférences du décideur par une fonction dite "d'utilité" en utilisant différents outils mathématiques. Ces méthodes sont souvent de type agrégation, dans le sens où la fonction d'utilité résultante est souvent une agrégation de plusieurs "sous-fonctions" qui peuvent être les fonctions objectifs de chaque critère, ou une combinaison de deux ou plusieurs d'entre elles. Mathématiquement parlant, les méthodes MAUT tentent de trouver une fonction u telle que :

$$u(\mathbf{x}) = f(u_1(f_1(\mathbf{x})), u_2(f_2(\mathbf{x})), ..., u_k(f_k(\mathbf{x})))$$

où les fonctions u_i représentent des fonctions d'utilité mono-dimensionnelle de chaque objectif f_i. Il existe différentes méthodes MAUT, selon le principe de définition des fonctions d'utilité mono-dimensionnelles u_i ainsi que la fonction d'agrégation f. Par exemple, dans le cas d'un problème multiobjectf à deux dimensions, nous pouvons citer les fonctions d'utilité suivantes.

$$u(\mathbf{x}) = \alpha_1 u_1(f_1(\mathbf{x})) + \alpha_2 u_2(f_2(\mathbf{x})), \text{ tel que } \alpha_1 > 0 \ , \ \alpha_2 > 0$$

$$u(\mathbf{x}) = u_1(f_1(\mathbf{x}))^{\alpha} u_2(f_2(\mathbf{x}))^{\beta}, \text{ tel que } \alpha > 0 \ , \ \beta > 0$$

$$u(\mathbf{x}) = \alpha_1 u_1(f_1(\mathbf{x})) + \alpha_2 u_2(f_2(\mathbf{x})) + \alpha_3 (u_1(f_1(\mathbf{x})) - k_1)^{\alpha} (u_2(f_2(\mathbf{x})) - k_2)^{\beta}$$

Une fois que la fonction d'utilité est définie, les différentes actions peuvent être comparées au moyen des expressions suivantes.
Soit $\mathbf{x}$ et $\mathbf{x}'$ deux actions à comparer :

$$\mathbf{x} \text{ et } \mathbf{x}' \text{ sont "indifférents" } \Leftrightarrow u(\mathbf{x}) = u(\mathbf{x}')$$

$$\mathbf{x} \text{ "est préférable à" } \mathbf{x}' \Leftrightarrow u(\mathbf{x}) > u(\mathbf{x}')$$

En utilisant les expressions ci-dessus, le problème introduit dans la section 2.2.4 est ramené à un problème d'optimisation mono-objectif, défini comme suit :

$$\begin{cases} \text{Minimiser} : u(\mathbf{x}) \\ \text{s.t. } g_j(\mathbf{x}) \leq 0; j = 1, 2, ..., m \end{cases}$$

2.4.3 Méthode des métriques pondérées

La méthode des métriques pondérées (*Method of Weighted Metrics*) sélectionne un vecteur de critères qui minimise une distance à une solution de référence $\mathbf{R} = (R_1, ..., R_k)$. Le vecteur $\mathbf{R}$ est soit le point idéal, soit une solution de référence fixée par le décideur en fonction de ses préférences [Mie99]. Mathématiquement parlant, la méthode consiste à minimiser une distance Δ_p définie comme suit :

$$\Delta_p(\mathbf{x}) = \left(\sum_1^k \omega_i \left(R_i - f_i(\mathbf{x}) \right)^p \right)^{\frac{1}{p}}$$

Selon la valeur de p, des cas particuliers de distances bien connues peuvent être utilisées :

1. p = 1 : la distance de Manhattan pondérée

$$\Delta_1(\mathbf{x}) = \sum_1^k \omega_i \left| R_i - f_i(\mathbf{x}) \right|$$

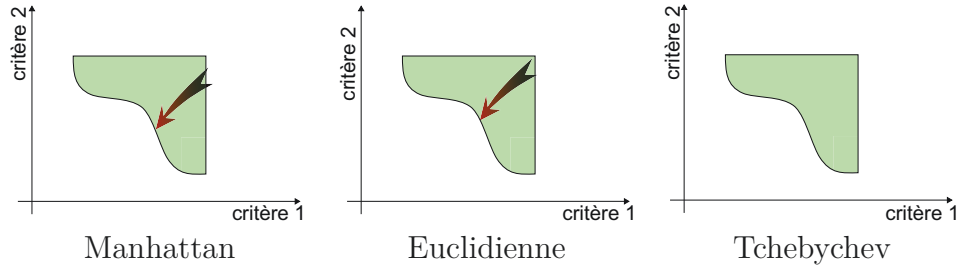
2. p = 2 : la distance euclidienne pondérée

$$\Delta_2(\mathbf{x}) = \sqrt{\left(\sum_1^k \omega_i \left(R_i - f_i(\mathbf{x}) \right)^2 \right)}$$

3. p = ∞ : la distance de Tchebychev pondérée

$$\Delta_\infty(\mathbf{x}) = \max_{i \in [1,k]} \omega_i \left| R_i - f_i(\mathbf{x}) \right|$$

La figure 2.4 donne une représentation graphique des trois distances citées ci-dessus. Du point de vue des relations entre critères, la distance de Manhattan correspond au cas où une compensation entre critères est possible, c'est-à-dire qu'une augmentation de la distance Δ_1 en raison d'une détérioration d'un critère donné peut être compensée



Manhattan Euclidienne Tchebychev

FIGURE 2.4 – Différents types de distances

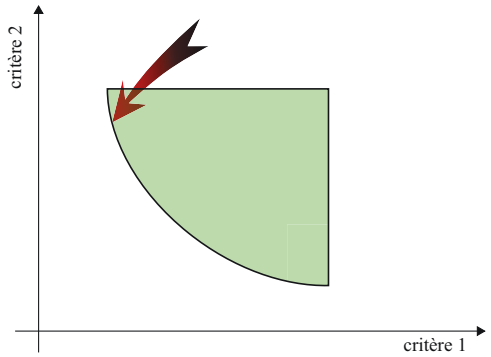


FIGURE 2.5 – Problématique du choix du point de référence

par une amélioration des autres critères. Quant à la distance de Tchebychev, elle correspond à une situation où aucune compensation n'est possible entre critères.

La méthode des métriques pondérées utilise aussi des poids de pondération ω_i, ce qui la place au niveau de la méthode de la somme pondérée relativement à la difficulté de traduire les préférences de l'utilisateur par des poids. De plus, la méthode est très sensible à la position du point de référence dans l'espace des critères. Par exemple, la figure 2.5 montre une situation où le point de référence 1 engendre une solution qui minimise uniquement le critère 1 et le point de référence 2 qui minimise uniquement le critère 2. Cela correspond à une optimisation monocritère. Ceci dit, la méthode des métriques pondérées dépend fortement à la fois du choix du point de référence et des valeurs des poids utilisés dans l'expression de la distance. Ce qui rend difficile son utilisation dans un contexte de système autonome.

2.4.4 Méthode de programmation par buts

La méthode de programmation par buts (*Goal programming*) [CC61] est une extension de la méthode de la programmation linéaire au cas multiobjectif. Cette méthode se caractérise par la fixation d'un but à atteindre pour chaque critère du problème. La méthode consiste à minimiser les déviations de chaque critère par rapport à son but. Mathématiquement parlant, la méthode de programmation par buts transforme le problème décrit dans la section 2.1 comme suit.

Premièrement, on définit k valeurs réelles $F_1, ..., F_k$ qui représentent les buts à atteindre pour les fonctions objectifs $f_1, ..., f_k$ respectivement. Ensuite, on crée $2 \times k$ variables $\delta_1^+, \delta_1^-, ..., \delta_k^+, \delta_k^-$, appelées variables de déviation, qui représentent le degré de déviation de la solution par rapport aux fonctions objectifs $f_1, ..., f_k$ respectivement. Finalement, on obtient le problème suivant :

$$\begin{cases} \text{Minimiser} : (\delta_1^+ \text{ ou } \delta_1^-, ..., \delta_k^+ \text{ ou } \delta_k^-) \\ f_1(\mathbf{x}) = F_1 + \delta_1^+ - \delta_1^- \\ ... \\ f_k(\mathbf{x}) = F_k + \delta_k^+ - \delta_k^- \\ g_j(\mathbf{x}) \leq 0; j = 1, 2, ..., m \end{cases}$$

Les variables de déviation δ_i^+ et δ_i^- doivent vérifier les conditions suivantes :

δ_i^+ et $\delta_i^- \geq 0$, et $\delta_i^+.\delta_i^- = 0$ tel que $i \in \{1, ..., k\}$

La minimisation des variables δ_i^+ permet d'atteindre par valeurs supérieures les objectifs f_i, tandis que minimiser δ_i^- permet d'atteindre par valeurs inférieures les objectifs i.

Dans cette méthode, le problème de décision est ramené à la minimisation d'un vecteur de variables de déviation. Il est par conséquent primordial de définir une fonction d'agrégation des différentes composantes de ce dernier. Par exemple, si l'on désire approcher par valeurs supérieures les objectifs, nous pouvons minimiser la somme pondérée des variables δ_i^+. Bien évidemment, on tombe sur le problème difficile du choix des poids que nous avons évoqué dans la section 2.4.1.

2.4.5 Méthode AHP

La méthode *AHP* (*Analytic Hierarchy Process*) a été introduite par Saaty en 1980 [Saa80]. Cette méthode tient son avantage de sa similitude au mécanisme de décision de l'être humain, à savoir décomposition, jugement et synthèse. Elle se distingue notamment par sa capacité à gérer différentes classes de critères, à savoir les critères qualitatifs et quantitatifs. De nos jours, elle est très utilisée dans des domaines variés, incluant l'économie, l'écologie, l'industrie, etc. La méthode AHP opère en cinq phases principales :

1. Décomposition hiérarchique du problème de décision initiale : le premier niveau étant l'objectif global du problème, le dernier niveau est formé par les différentes alternatives (actions) offertes au décideur et les niveaux intermédiaires sont occupés par des sous-critères dérivés des critères des niveaux supérieurs.

2. Construction des matrices de comparaison deux à deux : dans cette phase, les sous-critères de chaque niveau, ou alors les alternatives du dernier niveau, sont comparés deux à deux en fonction de leur importance vis-à-vis de l'objectif englobant situé dans le niveau supérieur. Pour ce faire, la méthode AHP utilise un système de comparaison comportant une échelle allant de 1 (importance égale de deux critères) à 9 (un critère est absolument plus important qu'un autre), selon l'importance des sous critères vis-à-vis du critère englobant. Bien évidemment, c'est le décideur qui se charge de cette phase.

3. Établissement des priorités : les priorités de chaque niveau intermédiaire sont représentées par des poids. Elles sont obtenues à partir des principaux vecteurs propres des matrices formées dans la phase 2. Pour ce faire, les valeurs des composantes des principaux vecteurs propres de chaque matrice de comparaison deux à deux sont normalisées à 1. Une valeur proche de 1 signifie que l'élément correspondant est plus prioritaire vis-à-vis du critère englobant.

4. Synthèse des priorités : les priorités finales sont calculées de proche en proche en agrégeant par une somme pondérée les priorités du niveau courant, pondérées par les priorités du niveau supérieur.

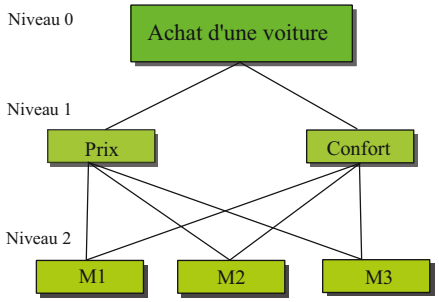


FIGURE 2.6 – Structure hiérarchique d'un problème de décision vu par la méthode AHP

5. Cohérence des jugements : la méthode AHP introduit un paramètre spécial, appelé "ratio de cohérence", qui est utilisé pour mesurer la cohérence des jugements du décideur (notamment dans la phase de comparaison mutuelle des éléments). En résumé, le ratio de cohérence peut être défini comme la probabilité qu'une matrice de comparaison deux à deux soit complétée aléatoirement.

Pour illustrer le fonctionnement de la méthode, prenons l'exemple de l'achat d'une voiture. L'objectif global du problème est l'achat d'une voiture. Nous considérons deux critères, à savoir le prix (P) et le confort (C). Enfin, nous supposons que le décideur sera amené à choisir entre trois modèles M1, M2 et M3. La figure 2.6 illustre la structure hiérarchique du problème.

Suivant les étapes de la méthode AHP, le décideur est sollicité pour fournir trois matrices de comparaison deux à deux : une matrice 2×2 de comparaison des deux critères prix et confort, montrant l'importance des deux vis-à-vis de l'objectif global du problème ; deux matrices 3×3 traduisant l'importance des différents modèles M1, M2 et M3 vis-à-vis des deux critères prix et confort pris séparément. Nous supposons que le décideur a fourni les valeurs décrites dans les tableaux 2.1, 2.2 et 2.3. Dans le tableau 2.1, la valeur 5 signifie que le prix est 5 fois plus important que le confort, ce qui justifie la valeur $\frac{1}{5}$ ligne C colonne P. L'intensité d'importance est donnée par la table établie par Saaty en fonction d'un jugement formulé par le décideur [Saa08].

Comme énoncé ci-dessus, le vecteur des priorités (dernière colonne dans les tableaux 2.1, 2.2 et 2.3) correspond au vecteur propre associé à la valeur propre maximale de chaque matrice de comparaison deux à deux. Par exemple et pour la matrice 2.2, la

	P	C	priorité
P	1	5	0,83
C	$\frac{1}{5}$	1	0,17

TABLE 2.1 – Matrice de comparaison deux à deux du niveau 1

	M1	M1	M3	priorité
M1	1	3	7	0,6586
M2	$\frac{1}{3}$	1	4	0,2628
M3	$\frac{1}{7}$	$\frac{1}{4}$	1	0,0786

TABLE 2.2 – Matrice de comparaison deux à deux du niveau 2 (critère Prix)

valeur propre maximale est $\lambda_{max} = 3,0324$ et le vecteur propre correspondant est $W = (0,9232; 0,3683; 0,1102)$. Enfin, le vecteur des priorités $W_p = (0,6586; 0,2628; 0,0786)$ est obtenu en divisant chaque coordonnée par la somme des valeurs de toutes les coordonnées.

La consistance des matrices de comparaison deux à deux est déterminée au moyen du "ratio de consistance" CR (*Consistency Ratio*). Le ratio de consistance est le rapport de l'indice de consistance CI (*Concistency Index*) à l'indice de consistance aléatoire RI (*Random Index*).

$$CR = \frac{CI}{RI}$$

tel que,

$$CI = \frac{\lambda_{max} - n}{n - 1}$$

où n est le nombre de lignes ou colonnes de la matrice de comparaison deux à deux. Concernant le paramètre RI, Saaty a fourni dans son livre [Saa80] des valeurs pour des matrices de comparaison deux à deux de tailles différentes. Par exemple, pour $n = 3$, $RI = 0,58$. Par conséquent, le ratio de consistance de la matrice 2.2 est obtenu comme suit :

$$CI = \frac{3,0324 - 3}{3 - 1} = 0,0162$$

	M1	M2	M3	priorité
M1	1	2	4	0,5714
M2	$\frac{1}{2}$	1	2	0,2857
M3	$\frac{1}{4}$	$\frac{1}{2}$	1	0,1429

TABLE 2.3 – Matrice de comparaison deux à deux du niveau 2 (critère Confort)

$$CR = \frac{0,0162}{0,58} = 0,028$$

Selon Saaty [Saa80], une matrice de comparaison deux à deux est consistante si $RC < 0,1$. De ce fait, la matrice 2.2 est consistante. Notons que le ratio de consistance doit être calculé pour chaque matrice de comparaison deux à deux afin de vérifier leur consistance. Dans le cas où une inconsistance est détectée, le décideur sera amené à reformuler ses jugements.

Une fois les priorités obtenues, les scores finaux des alternatives sont calculés en faisant une synthèse des scores partiels dans les différents niveaux. Les scores finaux de notre exemple sont donnés comme suit :

$$M1 = 0,83 \times 65,86\% + 0,17 \times 57,14\% = 64,38\%$$

$$M2 = 0,83 \times 26,28\% + 0,17 \times 28,57\% = 26,67\%$$

$$M3 = 0,83 \times 7,86\% + 0,17 \times 14,29\% = 8,95\%$$

Les résultats de cet exemple montrent que le modèle M1 est meilleur que le modèle M2, qui est meilleur que le modèle M3. Ceci est valable pour le décideur qui a fourni les matrices de comparaison deux à deux décrites dans les tableaux 2.1, 2.2 et 2.3.

Notons que cet exemple comprend uniquement deux niveaux. Dans le cas où la structure hiérarchique du problème comprend plusieurs niveaux, les synthèses sont déterminées en séquence, à commencer par le niveau supérieur jusqu'au niveau le plus bas (niveau des actions).

Bien que le processus de décision incarné par la méthode AHP soit intuitif et hiérarchique, il est néanmoins difficile de l'utiliser dans un contexte de système autonome. Ceci est dû principalement à la difficulté de faire intervenir le décideur quand il s'agit

de fournir les matrices de comparaison deux à deux. De plus, il est impossible de faire intervenir le décideur dans des environnements complexes, qui nécessitent des prises de décision en temps réel.

2.5 Méthodes de surclassement

Les méthodes de surclassement permettent de prendre en compte l'incomparabilité de certains couples d'actions. La comparaison des couples d'actions permet de construire un graphe des relations entre actions. Ce graphe permet de prendre en compte d'éventuelles relations non transitives, telles que le paradoxe de *Condorcet*. Ce dernier se manifeste, par exemple, quand un ensemble de votants est amené à choisir entre trois actions A, B et C. Dans ce cas, une majorité des votants préfèrent l'action A sur B, une autre majorité préfèrent l'action B sur C et enfin, une autre majorité des votants préfèrent l'action C sur A.

Les méthodes de surclassement permettent de sélectionner une et une seule solution si et seulement si la relation est un ordre total, et un ensemble de solutions dans le cas contraire. Par ailleurs, ce type de méthode est applicable uniquement au cas de problèmes MADM.

Les méthodes de surclassement sont basées sur les comparaisons d'actions deux à deux en vue d'établir une relation de surclassement S dans un ensemble d'actions ou d'alternatives $\mathbf{X}$. Selon B. Roy, une relation de surclassement S est une relation binaire définie dans l'ensemble des actions $\mathbf{X}$, telle que :

$\mathbf{x}S\mathbf{x}'$ si et seulement si il y a "suffisamment d'arguments pour admettre que $\mathbf{x}$ est au moins aussi bonne que $\mathbf{x}'$" et "il n'y a pas de raison importante de prétendre le contraire". De cette définition découlent deux concepts :

- Concordance : pour qu'une action $\mathbf{x}$ surclasse une autre action $\mathbf{x}'$ il faut que la majorité (au sens large) des critères dans $\mathbf{x}$ soient au moins aussi bons que leurs correspondants dans $\mathbf{x}'$.
- Discordance : pour qu'une action $\mathbf{x}$ surclasse une autre action $\mathbf{x}'$ il faut que $\mathbf{x}'$ ne présente pas d'aspects, non négligeables, pour lesquels son avantage sur $\mathbf{x}$ est substantiel.

Dans les sections suivantes, nous présentons les méthodes de surclassement les plus abordées dans la littérature, à savoir les méthodes ELECTRE et PROMETHEE.

2.5.1 Méthode ELECTRE

Les méthodes ELECTRE (*ELimination Et Choix Traduisant la REalité*) sont issues des travaux de B. Roy et son équipe de recherche. La première méthode ELECTRE I a été introduite par B. Roy en 1968 [Roy68]. Dans cette section nous présenterons la méthode ELECTRE I en détail, puis nous donnerons un bref aperçu des méthodes ultérieures. Pour un exposé détaillé voir [CS02, RB93, Vin92].

Comme nous l'avons énoncé ci-dessus la méthode ELECTRE se base sur les deux concepts de concordance et discordance afin de classer les actions. Pour ce faire la méthode introduit deux paramètres appelés indice de concordance $C(\mathbf{x}, \mathbf{x}')$ et indice de non-discordance $D(\mathbf{x}, \mathbf{x}')$ définis comme suit :

Soit $\omega_i, i \in [1, k]$ les poids relatifs aux fonctions objectifs f_i dans le problème de décision. Ces poids sont fournis directement par le décideur.
Soit $\mathbf{x}$ et $\mathbf{x}'$ deux actions à comparer :

$$C(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{k} \delta_i(\mathbf{x}, \mathbf{x}')$$

Tel que :

$$\delta_i(\mathbf{x}, \mathbf{x}') = \begin{cases} \omega_i & \text{si} \quad f_i(\mathbf{x}) \geq f_i(\mathbf{x}') \\ 0 & \text{sinon} \end{cases}$$

Par construction, comme les poids sont normalisés, $C(x, x')$ varie entre 0 (x est dominé par x') et 1 (x domine x').

$$D(\mathbf{x}, \mathbf{x}') = \begin{cases} \frac{1}{t_i} \max_i(f_i(\mathbf{x}') - f_i(\mathbf{x})) & \text{si} \quad C(\mathbf{x}, \mathbf{x}') \neq 1 \\ 0 & \text{sinon} \end{cases}$$

Tel que :

t_i est l'amplitude de l'échelle associée à la fonction objectif f_i. De ce fait, on a $0 \leq D(\mathbf{x}, \mathbf{x}') \leq 1$.

Une fois les deux indices définis, les deux actions $\mathbf{x}$ et $\mathbf{x}'$ peuvent être comparées

comme suit :

$$\mathbf{x}S\mathbf{x}' \Longleftrightarrow \begin{cases} C(\mathbf{x},\mathbf{x}') \geq c \\ D(\mathbf{x},\mathbf{x}') \leq d \end{cases}$$

où, c et d sont les seuils de concordance et de discordance, respectivement. Ce sont des paramètres internes à la méthode. L'expression ci-dessus permet d'extraire de l'ensemble initial des actions un sous-ensemble d'actions qui ne sont surclassées par aucune autre.

Bien que le processus de décision avec la méthode ELECTRE I formalise bien celui du raisonnement humain, la méthode ELECTRE I a l'inconvénient, néanmoins, d'utiliser des poids de quantification de l'importance des différents critères. Par conséquent, malgré l'apport de cette méthode, le décideur est toujours confronté à la tâche difficile de fournir des poids de quantification.

La méthode ELECTRE I est à l'origine de plusieurs autres méthodes, dont le principe fondamental est celui de ELECTRE I, avec quelques extensions.

- ELECTRE II [RB73] : cette méthode remplace la relation de surclassement classique par deux nouvelles relations, à savoir le surclassement fort et le surclassement faible.
- ELECTRE III [Roy78] : cette méthode a introduit la notion de pseudo-critères qui remplacent les critères classiques. Les pseudo-critères sont modélisés par des fonctions dont l'expression est proche des fonctions d'appartenance connues dans le domaine de la logique floue.
- ELECTRE IV [RB93] : cette méthode se distingue par sa capacité à se passer des poids associés à chaque critère. Cependant, cet avantage est tempéré par la nécessité de déterminer un "degré de crédibilité" associé à chaque relation de surclassement utilisée par la méthode.

2.5.2 Méthode PROMETHEE

La méthode PROMETHEE (*Preference Ranking Organisation Method for Enrichment Evaluations*) [BMV78] est semblable à la méthode ELECTRE, dans le sens où elle

vise à construire une relation de surclassement dans l'espace des actions ; par contre, elle en diffère par la nature de cette relation. En effet, contrairement à la relation de surclassement construite par la méthode ELECTRE, qui est purement binaire, dans le sens où une relation surclasse ou non une autre, la relation construite par la méthode PROMETHEE est une relation de surclassement valuée. En d'autres termes, une action surclasse une autre avec une intensité de préférence numérique. De ce fait, la méthode PROMETHEE reprend presque les principes de la théorie de l'utilité multi-attributs, qui associe à une action une valeur d'utilité numérique. La méthode PROMETHEE fonctionne comme suit :

Premièrement, la méthode calcule l'intensité de préférence P_i d'une action $\mathbf{x}$ vis-à-vis d'une autre action $\mathbf{x}'$ sur la fonction objectif f_i à l'aide de la relation suivante :

$$P_i(\mathbf{x}, \mathbf{x}') = G(d)$$

tel que :

$$d = f_i(\mathbf{x}) - f_i(\mathbf{x}')$$

$G(d)$ est une fonction croissante pour $d \geq 0$ et nulle pour $d < 0$. Par conséquent, une valeur proche de 1 de $P_i(\mathbf{x}, \mathbf{x}')$ signifie que $\mathbf{x}$ est nettement préférable à l'action $\mathbf{x}'$ par rapport à la fonction objectif f_i. L'inverse est vrai si $P_i(\mathbf{x}, \mathbf{x}')$ est proche de 0. La fonction G peut prendre différentes formes et dépendre de certains paramètres au choix du décideur. Le tableau 2.4 montre des formes usuelles de cette fonction. Les paramètres p et q introduits dans les figures sont, respectivement, les seuils d'indifférence et de préférence stricte.

Comme dans ELECTRE I, la méthode PROMETHEE utilise aussi des poids ω_i pour quantifier l'importance des critères. Les poids sont utilisés pour calculer une moyenne pondérée des intensités de préférence $P_i(\mathbf{x}, \mathbf{x}')$, appelée "indicateur de préférence", comme suit :

$$\pi(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{k} \omega_i P_i(\mathbf{x}, \mathbf{x}')$$

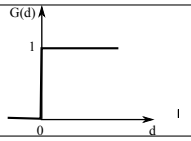
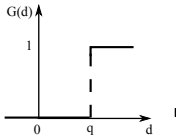
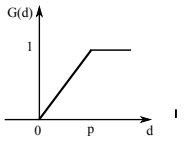
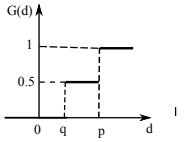
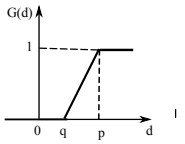
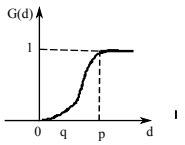
Critère	expression analytique	représentation graphique
Critère usuel	$G(d) = \begin{cases} 0 & \text{si } x = 0 \\ 1 & \text{si } x \neq 0 \end{cases}$	
Quasi-Critère	$G(d) = \begin{cases} 0 & \text{si } 0 \leq x \leq q \\ 1 & \text{si } x > q \end{cases}$	
Critère à préférence linéaire	$G(d) = \begin{cases} \frac{x}{p} & \text{si } 0 \leq x \leq p \\ 1 & \text{si } x > p \end{cases}$	
Critère à niveaux	$G(d) = \begin{cases} 0 & \text{si } 0 \leq d \leq q \\ \frac{1}{2} & \text{si } q < d \leq p \\ 1 & \text{si } d > p \end{cases}$	
Critère linéaire et zone d'indifférence	$G(d) = \begin{cases} 0 & \text{si } 0 \leq d \leq q \\ \frac{d-q}{p-q} & \text{si } q < d \leq p \\ 1 & \text{si } d > p \end{cases}$	
Critère gaussien	$1 - exp\left(-\frac{d^2}{2\sigma^2}\right)$	

TABLE 2.4 – Différentes formes de critères dans la méthode PROMETHEE

Ainsi, $\pi(\mathbf{x}, \mathbf{x}') \to 0$ signifie une faible préférence globale de $\mathbf{x}$ par rapport à $\mathbf{x}'$, et $\pi(\mathbf{x}, \mathbf{x}') \to 1$ traduit une préférence globale forte de $\mathbf{x}$ sur $\mathbf{x}'$.

L'ensemble des indicateurs de préférence est représenté par un graphe valué complet appelé "graphe de surclassement". Ce graphe est utilisé pour calculer les flux entrant ϕ^- et sortant ϕ^+ comme suit :

$$\phi^-(\mathbf{x}') = \sum_{\mathbf{x} \in \mathbf{X}} \pi(\mathbf{x}, \mathbf{x}')$$

$$\phi^+(\mathbf{x}') = \sum_{\mathbf{x} \in \mathbf{X}} \pi(\mathbf{x}', \mathbf{x})$$

Les flux entrant et sortant représentent les intensités des avantages et des désavantages qu'une action $\mathbf{x}'$ possède vis-à-vis de l'ensemble des autres actions candidates. Les flux entrant et sortant sont utilisés différemment par les méthodes PROMETHEE I et PROMETHEE II afin de définir des relations de pré-ordre.

– PROMETHEE I

  La méthode PROMETHEE I propose un pré-ordre partiel (permet l'incomparabilité) défini comme suit :

$$\begin{cases} \mathbf{x} P \mathbf{x}' \text{ si } \begin{cases} \phi^+(\mathbf{x}) > \phi^+(\mathbf{x}') \text{ et } \phi^-(\mathbf{x}) < \phi^-(\mathbf{x}') \\ \text{ou } \phi^+(\mathbf{x}) > \phi^+(\mathbf{x}') \text{ et } \phi^-(\mathbf{x}) = \phi^-(\mathbf{x}') \\ \text{ou } \phi^+(\mathbf{x}) = \phi^+(\mathbf{x}') \text{ et } \phi^-(\mathbf{x}) < \phi^-(\mathbf{x}') \end{cases} \\ \mathbf{x} I \mathbf{x}' \text{ si } \phi^+(\mathbf{x}) = \phi^+(\mathbf{x}') \text{ et } \phi^-(\mathbf{x}) = \phi^-(\mathbf{x}') \\ \mathbf{x} R \mathbf{x}' \text{ sinon} \end{cases}$$

  où les notations P, I et R signifient Préférence, Indifférence et Incomparabilité, respectivement.

– PROMETHEE II

  La méthode PROMETHEE II permet de construire un pré-ordre total (exclut l'incomparabilité et réduit considérablement l'indifférence) défini comme suit :

Soit : $\phi(\mathbf{x}) = \phi^+(\mathbf{x}) - \phi^-(\mathbf{x})$

$$\begin{cases} \mathbf{x} P \mathbf{x}' \text{ si } \phi(\mathbf{x}) > \phi(\mathbf{x}') \\ \mathbf{x} I \mathbf{x}' \text{ si } \phi(\mathbf{x}) = \phi(\mathbf{x}') \end{cases}$$

On remarque que la méthode PROMETHEE II se rapproche nettement des méthodes d'utilité, la mesure $\phi(\mathbf{x})$ étant l'utilité accordée à l'action $\mathbf{x}$. Par contre, la méthode PROMETHEE I garde toujours sa filiation aux méthodes de surclassement, grâce aux flux entrant et sortant qui sont relativement similaires aux notions de concordance et de discordance utilisées dans les méthodes ELECTRE.

2.6 Méthodes évolutionnaires

Les méthodes évolutionnaires, principalement les méthodes basées sur les algorithmes génétiques (AG), s'inscrivent dans la famille des méthodes *a posteriori*, dans le sens où le but est de constituer un ensemble de solutions non-dominées qui seront soumises ensuite aux préférences du décideur. Dans cette section, nous présentons brièvement la notion d'algorithme génétique, qui constitue le coeur de ces méthodes, puis nous donnerons un bref aperçu de la version multiobjectif de ce dernier. Pour une présentation détaillée, voir [DPST03].

2.6.1 Algorithmes génétiques

Les AGs ont été introduits par Holland en 1975. Ils sont inspirés de la théorie de l'évolution des espèces vivantes. La théorie de l'évolution stipule que les êtres vivants sont constamment soumis au mécanisme de compétition qui sélectionne les individus les plus adaptés à leur environnement. De ce fait, les individus les plus robustes (plus adaptés à leur environnement) auront plus de chance de se reproduire et de transmettre ainsi leurs gènes favorables, qui ont permis leur survie, à leurs enfants. Par conséquent, au fil des générations, les individus qui portent une combinaison de gènes adéquate formeront la majorité au sein de la population. De plus, de nouveaux individus peuvent apparaître au cours de l'évolution, grâce à la variation aléatoire des gènes, ce qui peut

enrichir davantage la population avec de forts individus. Les AGs utilisent les concepts suivants, afin de résoudre des problèmes d'optimisation complexes :

- Individu : représente une solution potentielle au problème d'optimisation. Un individu est composé de plusieurs gènes. Un gène est codé par un bit (0 ou 1) ou un entier ou encore un nombre réel, selon la représentation choisie pour traiter le problème ;
- Génotype : ensemble des caractéristiques héréditaires d'un individu ;
- Population : l'ensemble des individus traités simultanément par l'algorithme génétique ;
- Fonction d'adaptation (*fitness*) : une fonction qui associe à chaque individu une valeur numérique représentant sa qualité ;
- Génération : une itération de l'AG qui enchaîne l'application sur la population des opérateurs de sélection, mutation, croisement et remplacement décrits ci-dessous ;
- Sélection : associe à chaque individu une probabilité de reproduction à chaque génération. La probabilité de reproduction est souvent corrélée avec la qualité de l'individu. Ainsi, les meilleurs individus ont de fortes chances d'être sélectionnés pour se reproduire ;
- Remplacement : sélectionne l'ensemble des individus qui seront conservés au début d'une nouvelle génération ;
- Mutation : c'est un opérateur de variation qui modifie aléatoirement les gènes d'un individu ;
- Croisement : c'est un opérateur de variation qui combine un ou plusieurs individus parents pour former plusieurs individus enfants ;
- Élitisme : c'est un opérateur qui permet de préserver les meilleurs individus de la population à travers les différentes générations ;
- Nichage : opérateur qui permet de diversifier les individus au sein d'une population. En d'autres termes, l'opérateur de nichage permet d'éviter une situation où les copies d'un seul individu composent la majorité au sein de la population.

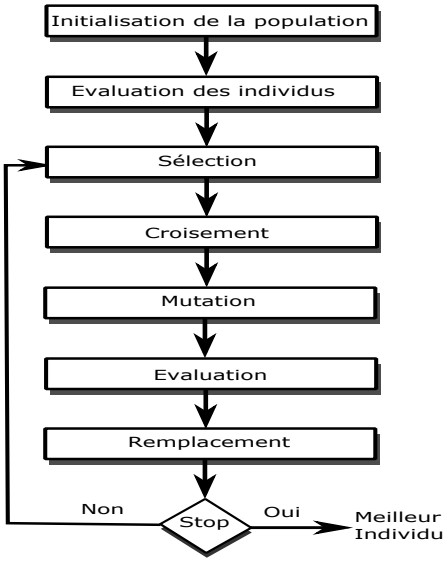


FIGURE 2.7 – Fonctionnement d'un algorithme génétique

La figure 2.7 illustre le schéma général de fonctionnement d'un algorithme génétique. D'abord, un ensemble de solutions ou individus (population initiale) est généré au début de l'optimisation. Généralement, les individus sont choisis par un tirage aléatoire au niveau de chaque gène. Ensuite, les individus sont évalués au moyen de la fonction d'adaptation (*fitness*) afin de déterminer leurs performances. Puis, l'algorithme génétique entre dans une boucle générationnelle (chaque itération étant une génération) dans laquelle les phases de sélection, croisement, mutation, évaluation et reproduction sont exécutées en séquence. Enfin, l'AG s'arrête en fonction d'un critère d'arrêt donné, tel qu'un nombre déterminé d'itérations, un temps donné d'exécution, stabilisation de la population, etc. A la fin de l'optimisation, le ou les meilleurs individus de la dernière génération sont retenus comme solutions du problème traité.

2.6.2 Algorithme génétique multiobjectif

Nous avons vu dans la sous-section précédente qu'un AG manipule plusieurs solutions à la fois afin de maximiser la probabilité de tomber sur l'optimum global du problème d'optimisation traité. Cette caractéristique est d'autant plus importante dans un contexte multiobjectif car elle permet à l'AG de bien échantillonner le front de Pareto. De nos jours, les AGs sont très utilisés pour résoudre les problèmes d'optimisation

multiobjectif. Des chercheurs publient constamment des travaux relatifs à ce domaine.

La différence principale entre un AG mono-objectif et un AG multiobjectif réside dans la phase d'évaluation de l'adaptation (*fitness*) des individus, particulièrement la manière de gérer les contradictions qui existent entre les différents objectifs. Dans ce contexte, on distingue deux approches principales, en l'occurrence l'approche élitiste et l'approche non-élitiste [Deb01]. Nous donnons, ci-dessous, un aperçu de ces deux approches.

Approches non-élitistes

L'algorithme VEGA [Sch85] (*Vector Evaluated Genetic Algorithm*) est la première tentative de concevoir un AG multiobjectif. VEGA exploite individuellement les fonctions objectifs partielles pour déterminer l'efficacité des individus. L'algorithme procède comme suit : premièrement, l'ensemble des individus de la population est partitionné en un nombre de groupes égal au nombre de fonctions objectifs partielles. Ensuite, on associe une fonction objectif partielle à chaque groupe ; par conséquent, l'efficacité d'un individu est déterminée par une seule fonction objectif. Ensuite, tous les individus sont mélangés, et les phases d'un AG classiques sont exécutées. Ce processus est répété tout au long des itérations d'évolution de l'AG.

L'algorithme VEGA souffre d'un biais engendré par les individus les plus forts sur chaque objectif individuel, ce qui concentre la recherche sur des zones particulières du front de Pareto. Pour remédier à ce problème, David E. Goldberg propose dans son livre [Gol89] d'utiliser la notion de dominance dans la fonction d'adaptation pour diriger la recherche vers le front de Pareto, et le mécanisme de nichage qui permet de répartir uniformément les individus sur toute la surface du front de Pareto. Cette contribution a permis le développement de plusieurs approches, qui diffèrent les unes des autres par la façon de formuler la fonction d'adaptation.

Dans [FF$^+$93], l'adaptation d'un individu est mesurée par son rang dans la liste de dominance. Le rang d'un individu est associé au nombre d'individus qui le dominent au sens de Pareto. Dans le cas d'une minimisation, on affecte un rang égal à 1 pour les individus non-dominés, puis des rangs supérieurs à 1 aux individus dominés. Ensuite,

les rangs sont ramenés à des valeurs d'efficacité en leur appliquant une interpolation à l'aide d'une fonction décroissante.

Dans [HNG94], les auteurs utilisent la notion de tournoi basée sur la dominance de Pareto pour sélectionner les individus dans la phase de remplacement de l'algorithme *Niched Pareto GA* (NPGA). D'abord, un ensemble d'individus (ensemble de comparaison) est sélectionné à partir de la population courante. Ensuite, une paire d'individus est choisie aléatoirement dans le reste de la population. Ces individus sont comparés avec les individus de l'ensemble de comparaison. Si l'un des individus domine et l'autre est dominé, alors l'individu dominant est sélectionné. Si les deux individus dominent ou sont dominés par les individus de l'ensemble de comparaison, alors l'individu qui est sélectionné est celui qui a le moins de voisins dans une boule de rayon donné dans l'espace des critères.

L'algorithme NSGA (*Non-dominated Sorting Genetic Algorithm*) introduit dans [SD94] calcule la fonction d'adaptation des individus comme suit : d'abord, l'ensemble de la population courante est trié pour trouver le premier sous-ensemble des individus non-dominés. L'adaptation partagée de ces individus est obtenue en appliquant la méthode de nichage dite "du partage" [GR87] avec une adaptation brute prise égale à la taille de la population. La méthode du partage nécessite la détermination d'un rayon de niche, ce qui est un paramètre difficile à déterminer. Ensuite, le second sous-ensemble des individus non-dominés est déterminé en ignorant les individus appartenant au premier sous-ensemble. Les individus du deuxième sous-ensemble auront une adaptation inférieure à l'adaptation partagée du plus mauvais individu du premier sous-ensemble des individus non-dominés. Ce processus est réitéré jusqu'à ce que tous les individus de la population se voient attribuer une valeur d'adaptation.

Approches élitistes

Le mécanisme d'élitisme permet à un AG de préserver les meilleurs individus rencontrés au fil des générations. Ce mécanisme est efficace pour diriger la recherche vers le front de Pareto. La seconde génération des algorithmes génétiques multiobjectif se base sur ce principe afin d'améliorer leurs performances. Nous décrivons brièvement,

dans cette section, deux approches élitistes très connues, en l'occurrence NSGA-II [DAPM00] et *Strength Pareto EA* (SPEA) [ZT98].

L'algorithme SPEA maintient les individus non-dominés découverts au fil des générations dans un ensemble particulier, appelé archive. Les individus de l'archive participent à l'évolution de la population et permettent, par conséquent, de diriger la recherche vers le front de Pareto. Au début de chaque itération de l'algorithme, les individus de l'archive sont mélangés avec la population courante. Ensuite, les adaptations des individus sont calculées comme suit : les individus non-dominés auront une adaptation égale au nombre d'individus qu'ils dominent, divisée par la taille de la population, plus un. Quant aux individus dominés, ils auront une adaptation égale à la somme des adaptations des individus qui les dominent, plus un. La fonction d'adaptation ainsi définie doit être minimisée. Ceci permet d'orienter la recherche vers les individus non-dominés. Dans [ZLT02], les auteurs proposent une amélioration de l'algorithme SPEA, appelée SPEA2, qui diffère de SPEA au niveau du calcul de la fonction d'adaptation.

NSGA-II est une amélioration de l'algorithme initial NSGA avec le principal objectif d'ajouter l'élitisme, qui améliore beaucoup les performances. De plus, La complexité de NSGA-II est de l'ordre de $O(mN^2)$, alors que celle de NSGA est de l'ordre de $O(mN^3)$, où N est la taille de la population et m est le nombre d'objectifs. Enfin, la méthode ne nécessite plus la détermination d'un rayon de niche.

2.6.3 Application aux systèmes autonomes

Les méthodes évolutionnaires sont difficiles à utiliser dans un contexte de systèmes autonomes car le but des AGs est de fournir un ensemble de solutions qui seront soumises au décideur, afin qu'il choisisse la solution appropriée selon ses préférences. Des approches récentes [DS06] permettent de lever une partie de cette problématique en combinant les algorithmes génétiques avec les méthodes de décision multicritère classiques, telles que celles qui utilisent un point de référence [Wie80]. Ces méthodes permettent de réduire l'ensemble des solutions proposées à une seule proposition qui s'accorde le mieux avec les préférences du décideur. Cependant, ces nouvelles approches héritent des problèmes des méthodes de décision multicritère classiques, notamment le

choix du point de référence et la détermination de la valeur des poids de quantification de l'importance des critères dans le processus de décision multicritère.

2.7 Conclusion

Il est souvent difficile de résoudre des problèmes de décision mono-critère, car ces problèmes issus du monde réel sont souvent de complexité élevée. Cette difficulté est accrue si plusieurs critères contradictoires sont associés au problème de décision. En effet, en plus de l'explosion combinatoire, il faut gérer les préférences du décideur afin de trouver une solution satisfaisante. Par sa complexité, le domaine de la décision multicritère a engendré une multitude de méthodes diverses selon leur principe d'incorporation des préférences du décideur dans le processus de décision.

Dans ce chapitre, nous avons considéré les éléments essentiels de la décision multicritère. Nous avons exposé les différentes méthodes qui essayent de trouver les solutions non-dominées d'un problème d'optimisation multiobjectif ou de sélectionner une solution qui s'accorde le mieux avec les préférences du décideur. Dans ce contexte, nous avons constaté que la majorité de ces méthodes utilisent des paramètres spécifiques afin de déterminer l'importance de chaque critère dans le processus de décision. Ces paramètres sont soit donnés directement par le décideur, soit obtenus par des matrices de comparaison deux à deux fournies aussi par le décideur. Par ailleurs, les méthodes évolutionnaires permettent d'obtenir un échantillon de solutions les plus proches possibles du front de Pareto. La meilleure solution est sélectionnée par le décideur. Par conséquent, ces procédés sont inadéquats dans un contexte autonome car le système est supposé fonctionner indépendamment d'un décideur.

Le chapitre suivant introduit une approche multicritère conçue pour un contexte autonome.

Chapitre 3

Une approche multicritère pour les systèmes autonomes

3.1 Introduction

Les méthodes de décision multicritère ou d'aide à la décision multicritère permettent de sélectionner un ensemble de solutions alternatives à un problème de décision multicritère. Ce travail doit être complété par le décideur, qui doit choisir la meilleure alternative qui correspond le mieux à ses préférences. Cependant, il est difficile d'estimer les paramètres de ces méthodes qui, souvent, utilisent des poids pour quantifier l'importance de chaque critère dans le processus de décision. De ce fait, quand les alternatives offertes par la méthode ne sont pas conformes aux préférences du décideur, une interaction entre la méthode et ce dernier est indispensable afin d'affiner le processus de décision.

Les systèmes autonomes sont des systèmes autoadaptatifs qui sont capables d'apprendre le comportement de leur environnement afin de s'auto-configurer et de prendre des décisions optimales, sans avoir recours à une aide externe. Par conséquent, l'interaction avec un décideur pour la prise de décision multicritère est impossible pour ce type de systèmes. Les méthodes de décision multicritère classiques ne sont pas adéquates dans ce contexte, car la majorité d'entre elles se basent sur l'interaction avec le décideur afin d'affiner le processus de décision.

Les systèmes dynamiques adaptent constamment leur état, ainsi que leurs paramètres, sous différentes contraintes temporelles. Le front de Pareto de ce type de systèmes peut varier considérablement en fonction du temps, ce qui rend difficile la prise de décision multicritère, particulièrement dans le cas où les contraintes de temps sont en dessous des temps de réaction de l'être humain. Par conséquent, il est difficile, voire impossible, de compter sur l'interaction avec le décideur afin d'adapter constamment les paramètres des méthodes de décision multicritère classiques.

Dans ce chapitre, nous présenterons une nouvelle méthode de décision multicritère adaptée aux systèmes autonomes, ainsi qu'aux systèmes dynamiques. Ce chapitre est organisé comme suit : premièrement, nous introduisons dans la section 3.2 le principe général de la "méthode de la poupée russe". Ensuite, une description détaillée, comportant des définitions formelles, ainsi que l'exposé des différentes variantes de la méthode, est donnée dans la section 3.3. Enfin, la section 3.4 conclut ce chapitre.

3.2 Principe général de la méthode

L'objectif de la méthode consiste à construire des fonctions d'utilité en fonction d'un ensemble de paramètres fournis par le concepteur du système autonome. Nous rappelons que le concepteur du système diffère d'un décideur, dans le sens où le concepteur agit au moment de la conception d'un système en fonction de données préétablies, contrairement au décideur, qui agit au moment de la prise de décision.

Il existe plusieurs méthodes qui permettent au concepteur de définir les paramètres de la méthode de décision multicritère. Par exemple, il est commun d'évaluer la qualité d'une application multimédia, telle que la voix sur IP VoIP (*Voice over IP*) au moyen d'un sondage d'opinion appelé MOS (*Mean Opinion Score*) [Uni96]. Le MOS consiste à réunir l'opinion d'un ensemble d'utilisateurs sur plusieurs passages d'une application donnée (des séquences d'une communication VoIP, par exemple). L'opinion consiste, par exemple, à attribuer une note comprise entre zéro et cinq, en fonction de la qualité perçue. Cinq étant la qualité excellente et zéro la plus mauvaise qualité. La figure 3.1, illustre les résultats qui peuvent être obtenus au moyen d'un MOS (notons que cette figure est donnée à titre illustratif seulement, puisqu'elle ne provient d'aucune expé-

rience réelle). Dans cet exemple, la qualité de l'application VoIP est évaluée en fonction de deux métriques, à savoir le délai et le taux de perte. Nous remarquons, à travers la figure, que la qualité de la VoIP varie entre "excellente", "bonne", "acceptable" et "mauvaise". Suivant cette observation, nous pouvons décomposer l'espace des critères à l'aide d'un ensemble de boîtes englobantes, semblables à des poupées russes (voir figure 3.1). La boîte numéro 3 est définie comme la boîte qui couvre l'ensemble de l'espace des critères. Selon les observations illustrées dans la figure 3.1, la boîte 0 englobe les vecteurs de critères traduisant une excellente qualité de la VoIP. Les vecteurs de critères qui sont dans la boîte 1 et n'appartiennent pas à la boîte 0 traduisent une qualité bonne. Les vecteurs de critères qui sont dans la boîte 2 et n'appartiennent pas à la boîte 1 traduisent une qualité acceptable. Enfin, les vecteurs de critères restants traduisent une qualité mauvaise.

La figure 3.2 illustre un front de Pareto dans un espace de recherche formé par les deux critères délai et taux de perte. Cette figure montre que les boîtes 0 et 1 ne contiennent aucune solution appartenant au front de Pareto. Par conséquent, il existe au mieux des solutions *acceptables* pour cet exemple de front de Pareto. De ce fait, la solution choisie est la plus proche par rapport à la boîte de bonne qualité (boîte 1), selon une distance $\Delta(\mathbf{x})$ ($\mathbf{x}$ est un vecteur de critères) qui est à définir. Cette distance est utilisée pour définir la fonction d'utilité représentée par la méthode de la poupée russe.

La distance $\Delta(\mathbf{x})$ est donnée dans l'ensemble des boîtes de qualité avec des valeurs allant de 0 à m, où m est l'index de la plus grande boîte qui englobe tout l'espace de critères. $\Delta(\mathbf{x}) = 0$ signifie que le vecteur $\mathbf{x}$ est dans la boîte 0 qui est la boîte de la meilleure qualité possible. Cette distance permet de définir une fonction d'utilité u, telle que $u(\mathbf{x}) = m - \Delta(\mathbf{x})$. De ce fait, la fonction d'utilité u est maximale pour tout $\mathbf{x}$ appartenant à la boîte 0.

Plusieurs formes de boîtes peuvent être utilisées afin de construire une fonction d'utilité u qui est capable d'approcher une relation de qualité $Q(\mathbf{R}^n, \mathbf{R})$, où $\mathbf{R}^n$ désigne l'ensemble des vecteurs de critères et $\mathbf{R}$ est l'ensemble de définition des qualités. Pour notre part, nous avons choisi de représenter une boîte par l'intersection d'un hyper-

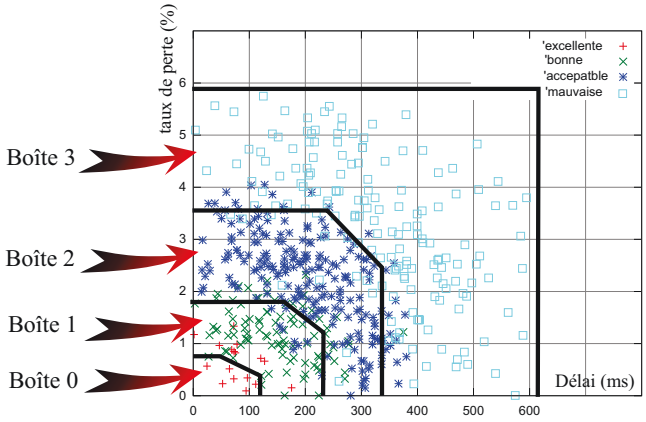


FIGURE 3.1 – Évaluation de la qualité de la VoIP en fonction du délai et du taux de perte

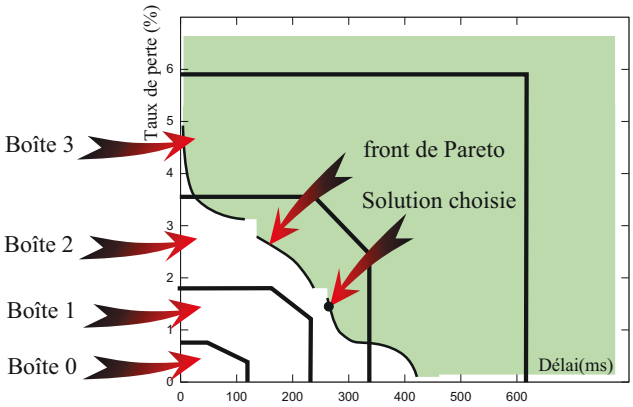


FIGURE 3.2 – Processus de décision via la méthode de la poupée russe

rectangle avec un simplexe. Ce choix est dû à des considérations que nous expliquerons dans la section 3.3.

Dans cette section, nous avons donné une idée générale de la méthode multicritère proposée. Les sections suivantes présentent en détail le fonctionnement de la méthode de la poupée russe.

3.3 Définition formelle de la méthode

Soit n le nombre de critères du problème. Sans perte de généralité, nous considérons, dans ce chapitre, le cas de la minimisation de tous les critères. De plus, nous supposons que tous les critères sont positifs ; dans le cas où un ou plusieurs critères ne le sont pas,

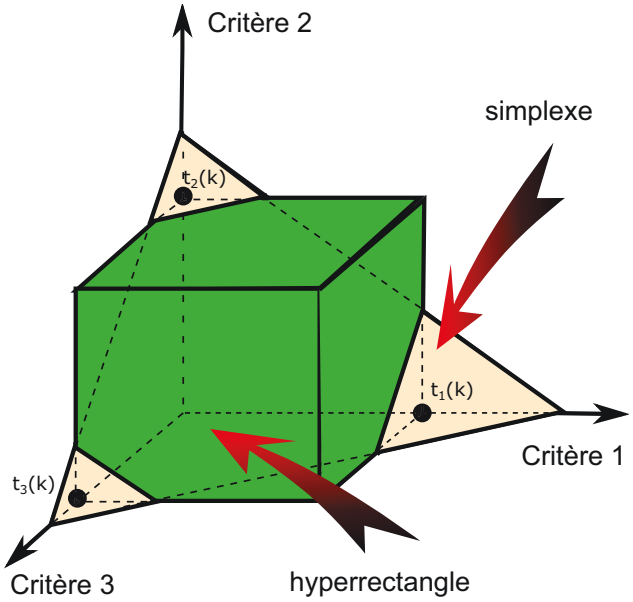


FIGURE 3.3 – Construction d'une boîte de qualité

une transformation affine est appliquée, afin de les ramener au quadrant positif. Un ensemble de $m+1$ boîtes indexées de 0 à m est défini dans l'espace de critères, de telle sorte que chaque boîte, excepté la boîte m, soit englobée dans une autre, comme une poupée russe (voir la figure 3.1). La boîte m, qui englobe tout l'espace des critères, est le produit cartésien de n intervalles $[0, u_i]$ tel que u_i est la valeur maximale du critère i. Chacune des m autres boîtes restantes est formée par l'intersection de $2n+1$ demi-espaces dont les bordures sont des hyperplans. Les hyperplans $\mathbf{H}_i\,(k)\,, 1 \leq i \leq n$ sont orthogonaux aux axes et l'hyperplan $\mathbf{H}_{n+1}\,(k)$ est oblique (voir la figure 3.3). Les hyperplans orthogonaux aux axes définissent un hyperrectangle et l'hyperplan oblique définit un simplexe. Ainsi, nous pouvons aussi définir une boîte par l'intersection d'un hyperrectangle avec un simplexe (voir la figure 3.3). Cette façon de construire une boîte de qualité permet d'obtenir une structure qui est capable de représenter différentes relations entre critères, notamment la compensation et la non-compensation (voir la figure 3.4). Les sous sections 3.3.3 et 3.3.4 présentent en détail ces deux types de relations.

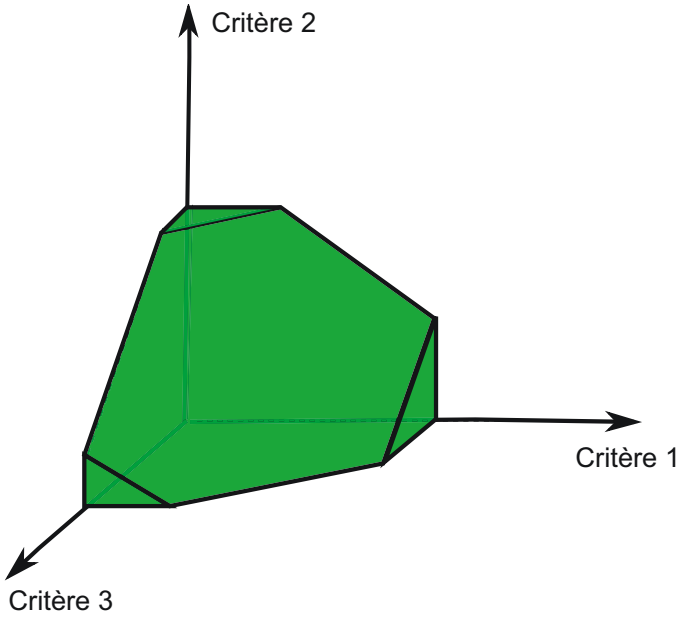


FIGURE 3.4 – Forme finale d'une boîte de qualité

Une boîte $\mathbf{B}(k)$ est définie par un ensemble de points $\mathbf{x} = (x_1, \ldots, x_n)^T, x_i \geq 0, \forall i \in [1, n]$ tel que :

$$\mathbf{A}(k)\,\mathbf{x} \leq \mathbf{b}$$

tel que $\mathbf{A}(k) = (a_{ij}(k))$ est une matrice $(n+1) \times n$ et $\mathbf{b}$ un vecteur colonne de taille $n+1$. "$\leq$" dans le système d'inéquations ci-dessus signifie que chaque élément de la matrice colonne $\mathbf{A}(k)\,\mathbf{x}$ est inférieur ou égal à son vis-à-vis dans $\mathbf{b}$. Dans ce qui suit, nous posons $\mathbf{b} = (1, \ldots, 1)$. Pour les hyperplans orthogonaux aux axes $\mathbf{H}_i(k), 1 \leq i \leq n : a_{ij}(k) = 0$ si $i \neq j$ et $a_{ii} > 0$. Pour l'hyperplan oblique $\mathbf{H}_{n+1}(k) : a_{n+1,j}(k) \geq 0$.

3.3.1 Distance globale d'un vecteur de critères à la boîte 0

Nous définissons la distance $\Delta(\mathbf{c})$ entre un vecteur de critères $\mathbf{c} = (c_1, \ldots, c_n)^T$ et la boîte $\mathbf{B}(0)$ comme suit :

$$\Delta(\mathbf{c}) = \begin{cases} k + \max_{i \in [1,n+1]} \delta_i(\mathbf{c}) & \text{si } \mathbf{c} \in \mathbf{B}(k+1) \text{ et } \mathbf{c} \notin \mathbf{B}(k) \\ 0 & \text{si } \mathbf{c} \in \mathbf{B}(0) \end{cases} \tag{3.1}$$

où, $\delta_i(\mathbf{c})$ est la distance normalisée entre le vecteur $\mathbf{c}$ et l'hyperplan $\mathbf{H}_i\,(k)$ représentant une frontière de la boîte $\mathbf{B}(k)$. Son expression est donnée ci-dessous.
Soit $\mathbf{p}$ la projection orthogonale du vecteur $\mathbf{c}$ sur l'hyperplan $\mathbf{H}_i\,(k)$. Soit $\mathbf{e}$ le point d'intersection de l'hyperplan $\mathbf{H}_h\,(k+1)$ composant la frontière supérieure de la boîte $\mathbf{B}(k+1)$ avec la normale de l'hyperplan $\mathbf{H}_i(k)$ de la boîte $\mathbf{B}\,(k)$ passant par p (Figure 3.5). L'hyperplan $\mathbf{H}_h\,(k+1)$ est choisi tel que $||\mathbf{p}-\mathbf{e}||$ est minimale, où $||\mathbf{x}||$ désigne le module du vecteur $\mathbf{x}$. D'abord, nous déterminons les coordonnées p_s et e_s, $s \in [1,n]$ en utilisant les expressions 3.2 et 3.3, respectivement, puis nous calculons la distance normalisée $\delta_i\,(\mathbf{c})$ en utilisant l'expression 3.4.

$$p_s = c_s + a_{is}(k)\frac{1-\sum_j^n a_{ij}(k)c_j}{\sum_{j=1}^n a_{i,j}^2(k)} \tag{3.2}$$

$$e_s = p_s + a_{is}(k)\frac{1-\sum_{j=1}^n a_{hj}(k+1)p_j}{\sum_{j=1}^n a_{hj}(k+1)a_{ij}(k)} \tag{3.3}$$

$$\delta_i(\mathbf{c}) = \begin{cases} 0 & \text{si } \sum_{j=1}^n a_{ij}(k)c_j \leq 1 \\ 0 & \text{si } \|\mathbf{p}-\mathbf{e}\| = 0 \\ \frac{\|\mathbf{p}-\mathbf{c}\|}{\|\mathbf{p}-\mathbf{e}\|} & \text{sinon} \end{cases} \tag{3.4}$$

où $\|\mathbf{x}-\mathbf{x}^{'}\|$ est la distance euclidienne entre $\mathbf{x}$ et $\mathbf{x}^{'}$.

Ainsi, chaque point situé sur la frontière supérieure de la boîte $\mathbf{B}\,(k+1)$ est à la distance 1 de la boîte englobée $\mathbf{B}\,(k)$, excepté dans le cas où $\|\mathbf{p},\mathbf{e}\| = 0$. Dans ce dernier cas, la distance vaut 0.

3.3.2 Processus de décision multicritère

Soient $\mathbf{u}$ et $\mathbf{v}$ deux vecteurs de critères à comparer. Le processus de décision multicritère est résumé dans l'algorithme suivant :

En appliquant l'algorithme ci-dessus, il est simple de démontrer que la relation "est meilleur que ou indifférent" est réflexive et transitive. Cependant, cette relation n'est

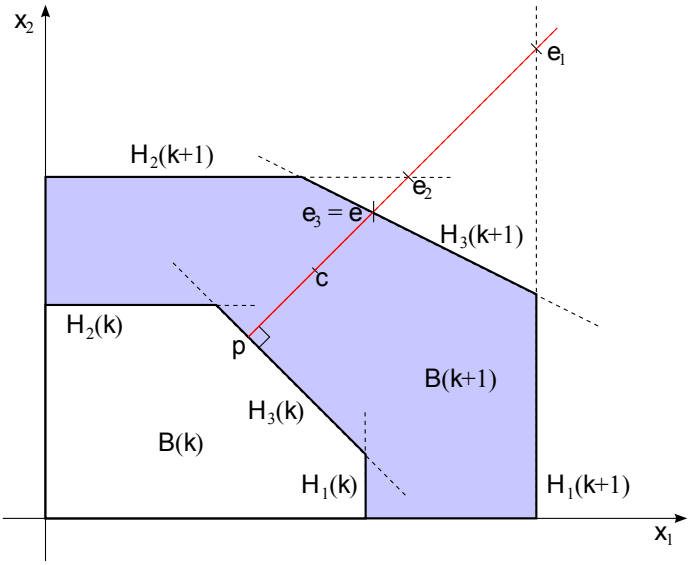


FIGURE 3.5 – Distance normalisée $\delta_3(\mathbf{c})$ entre l'hyperplan $\mathbf{H}_3(k)$ et $\mathbf{c}$

Algorithme 1: règles de comparaison

Données : deux vecteurs de critères $\mathbf{u}$ et $\mathbf{v}$
Résultat : décision sur la préférence
si $\Delta(\mathbf{u}) < \Delta(\mathbf{v})$ **alors**
  $\mathbf{u}$ est dit "meilleur que" $\mathbf{v}$;
sinon si $\Delta(\mathbf{u}) > \Delta(\mathbf{v})$ **alors**
  $\mathbf{v}$ est dit "meilleur que" $\mathbf{u}$;
sinon
  $\mathbf{u}$ et $\mathbf{v}$ sont dit indifférents";

pas antisymétrique, à cause de la règle “indifférent”, et elle n'est pas symétrique, à cause de la règle “meilleur que”.

La méthode de la poupée russe est implémentée en comparant les vecteurs de critères en utilisant les règles de comparaison explicitées ci-dessus. Le meilleur vecteur de critères est celui pour lequel la distance Δ est minimale.

3.3.3 Boîte de qualité hyper-rectangulaire

Une boîte de qualité donnée est définie par l'intersection d'un hyperrectangle avec un simplexe. Quand l'hyperrectangle est complètement à l'intérieur du simplexe, la boîte de qualité est réduite à un hyperrectangle. Dans ce cas, $a_{ij} = 0$ si $i \neq j$. De ce fait, la boîte de qualité est définie par les inéquations suivantes : $x_i \geq 0$, $a_{ii}(k)x_i \leq 1$, tel que : $1 \leq i \leq n$.
De plus, si nous considérons que toutes les boîtes qui composent la poupée russe sont des hyperrectangles, alors les équations (3.2) et (3.3) deviennent, respectivement :

$$p_s = \begin{cases} c_i & \text{si } i \neq s \\ 1/a_{ii}(k) & \text{sinon} \end{cases}$$

et

$$e_s = \begin{cases} c_i & \text{si } i \neq s \\ 1/a_{ii}(k+1) & \text{sinon} \end{cases}$$

Selon l'équation (3.4), pour tout vecteur de critères $\mathbf{x}$ choisi dans l'ensemble défini par l'intersection des boîtes $\mathbf{B}(k+1)$ et $\mathbf{B}(k)$, avec $a_{ii}(k) \neq a_{ii}(k+1)$:

$$\Delta(\mathbf{x}) = k + \max_{i \in [1,n]} \frac{x_i - p_i}{e_i - p_i}$$

Plus précisément, si $k = 0$ et $\mathbf{x}$ est choisi tel que $x_i \geq 1/a_{ii}(k)$, alors $\Delta(\mathbf{x})$ est une distance de *Chebyshev* pondérée entre $\mathbf{x}$ et $\mathbf{p} = (p_1, ..., p_i, ..., p_n)$.

Il est utile d'étudier le modèle d'agrégation utilisé pour résoudre un problème de décision multicritère du point de vue de la compensation ou la non-compensation entre critères. La compensation signifie qu'une diminution de la valeur de la fonction d'utilité en raison d'une détérioration d'un critère donné peut être compensée par une

Qualité	délai	gigue	taux de perte
Bonne	$[0, 150ms]$	$[0, 20ms]$	$[0, 0.5\%]$
Acceptable	$[150ms, 300ms]$	$[20ms, 50ms]$	$[0.5\%, 1.5\%]$
Mauvaise	$> 300ms$	$> 50ms$	$> 1.5\%$

TABLE 3.1 – Seuils de qualité des critères *délai*, *gigue* et *taux de perte* pour une application de vidéoconférence

amélioration des autres critères. La non-compensation entre critères signifie que la diminution de la valeur de la fonction d'utilité en raison d'un critère donné ne peut pas être compensée par une amélioration des autres critères.

La fonction d'utilité $u(\mathbf{x}) = m - \Delta(\mathbf{x})$ décroît quand $\Delta(\mathbf{x})$ croît. Cette situation survient lorsque x_i se détériore, alors que $(x_i - p_i)/(e_i - p_i)$ est maximal pour le critère i. Dans ce cas, l'amélioration des autres critères, c'est-à-dire la réduction de leur valeur, ne permet pas d'augmenter $u(\mathbf{x})$. Ainsi, une poupée russe composée de boîtes hyperrectangulaires forme un modèle d'agrégation non compensatoire. Un concepteur d'un système de décision multicritère peut vouloir utiliser un modèle non compensatoire lorsque certains critères sont incommensurables.

Nous définissons le seuil de qualité (QT : *Quality Threshold*) $t_i(k)$ comme la pire valeur du critère i pour laquelle la valeur de la qualité est k, sachant que les autres critères sont à leurs valeurs optimales. Typiquement, ce genre d'information devrait être facilement disponible pour les concepteurs du système. Ces paramètres pourraient provenir de normes, spécifications techniques des composants du système ou des expériences. Par exemple, pour une application de type vidéoconférence, ces paramètres peuvent être les données indiquées dans le tableau 3.1 [CSMS04]. Un hyperplan $\mathbf{H}_i(k)$ est défini par les coefficients $a_{ii}(k) = 1/t_i(k)$ et $a_{ij} = 0$ pour $i \neq j$ pour les autres coefficients. l'hyperplan oblique $\mathbf{H}_{n+1}(k)$ est inutile : par conséquent, il n'est pas défini.

3.3.4 Boîte de qualité de type simplexe

Quand le simplexe qui compose la boîte $\mathbf{B}(k)$ est complètement à l'intérieur de l'hyperrectangle, la boîte de qualité est réduite à un simplexe défini par les inéquations

suivantes :

$$\sum_{j=1}^{n} a_{n+1,j}(k)\, x_j \leq 1$$
$$x_j \geq 0$$

tel que : $\forall j \in [1,n], a_{n+1,j}(k) > 0$.

La distance globale à la boîte 0 devient :

$$\Delta(\mathbf{x}) = k + \delta_{n+1}(\mathbf{x})$$

Supposons que toutes les boîtes composant la poupée russe soient des simplexes. Nous voulons caractériser l'ensemble des points $\mathbf{x}$ qui sont à une distance $\delta_{n+1}(\mathbf{x}) = \gamma$ de l'hyperplan $\mathbf{H}_{n+1}(k)$, où γ est une constante choisie dans l'intervalle $[0,1]$. Le système de coordonnées est transformé par une translation et une rotation afin que l'équation de l'hyperplan $\sum_{j=1}^{n} a_{n+1,j}(k)\, x_j = 1$ dans le système de coordonnées initial devienne $x'_1 = 0$ dans le nouveau système de coordonnées. L'équation de l'hyperplan $\mathbf{H}_{n+1}(k+1)$ devient $\sum_{j=1}^{n} \alpha_{n+1,j} x'_j = 1$. Dans ce cas, tout point $\mathbf{x}'$ appartenant à l'hyperplan dont l'équation est $\alpha_{n+1,1} x'_1/\gamma + \sum_{j=2}^{n} \alpha_{n+1,j} x'_j = 1$ est à une distance $\delta_{n+1}(\mathbf{x}) = \gamma$ de l'hyperplan $\mathbf{H}_{n+1}(k)$. En effet :

Soit,

$$\mathbf{c} = \left(x_1^{'}, \ldots, x_n^{'}\right) \in \left(\frac{\alpha_{n+1,1} x_1^{'}}{\gamma} + \sum_{j=2}^{n} \alpha_{n+1,j} x_j^{'} = 1\right)$$

Donc,

$$\mathbf{c} = \left(\frac{\gamma\left(1 - \sum_{j=2}^{n} \alpha_{n+1,j} x_j^{'}\right)}{\alpha_{n+1,1}}, x_2^{'}, \ldots, x_n^{'}\right)$$

$\mathbf{p} = \left(0, x_2^{'}, \ldots, x_n^{'}\right)$ est la projection orthogonale du point $\mathbf{c}$ sur l'hyperplan dont l'équation est $x_1^{'} = 0$ et $\mathbf{e} = \left(\frac{1-\sum_{j=2}^{n} \alpha_{n+1,j} x_j^{'}}{\alpha_{n+1,1}}, x_2^{'}, \ldots, x_n^{'}\right)$.

Donc,

$$\|\mathbf{p} - \mathbf{c}\| = \sqrt{\gamma^2 \left(\frac{\sum_{j=2}^{n} \alpha_{n+1,j} x_j^{'} - 1}{\alpha_{n+1,1}}\right)^2}$$

Et

$$\|\mathbf{p}-\mathbf{e}\| = \sqrt{\left(\frac{\sum_{j=2}^{n}\alpha_{n+1,j}x_j^{'}-1}{\alpha_{n+1,1}}\right)^2}$$

Finalement,

$$\delta_{n+1}(\mathbf{c}) = \frac{\|\mathbf{p}-\mathbf{c}\|}{\|\mathbf{p}-\mathbf{e}\|} = \frac{\sqrt{\gamma^2\left(\frac{\sum_{j=2}^{n}\alpha_{n+1,j}x_j^{'}-1}{\alpha_{n+1,1}}\right)^2}}{\sqrt{\left(\frac{\sum_{j=2}^{n}\alpha_{n+1,j}x_j^{'}-1}{\alpha_{n+1,1}}\right)^2}} = \gamma$$

Par conséquent, les hypersurfaces iso-valeur dans une poupée russe de type simplexe sont des hyperplans.

Soit la distance $\delta_{n+1}(\mathbf{x})$ pour un vecteur de critères donné $\mathbf{x}$. Cette distance définit une hypersurface iso-valeur représentée par l'hyperplan $\mathbf{I}(\mathbf{x})$. La détérioration de la qualité d'un critère donné ne change pas la valeur de la distance, à condition que les autres critères soient choisis de telle sorte que le nouveau vecteur de critères appartienne aussi à l'hyperplan $\mathbf{I}(\mathbf{x})$. De ce fait, pour tout vecteur de critères $\mathbf{x}$ suffisamment loin des bords du simplexe, il existe un voisinage dans lequel la dégradation d'un critère peut toujours être compensée par l'amélioration des autres critères. En conclusion, la méthode de la poupée russe instanciée avec des boîtes simplexes est un modèle d'agrégation compensatoire.

Par ailleurs, soit $f(y)$ une fonction croissante de $y \in \mathbf{R}$ avec $f(0) = 1$. Si les coefficients $a_{n+1,j}(k) = a_{n+1,j}(0)/f(k), \forall j \in [1,n]$ pour toute boîte de qualité k, alors les hyperplans obliques qui forment les différentes boîtes simplexes sont parallèles. De ce fait, les hypersurfaces iso-valeur à la boîte 0 sont des hyperplans parallèles dont l'équation est $\sum_{j=1}^{n} a_{n+1,j}(0)\, x_j = f(\Delta(\mathbf{x}))$. Par conséquent, minimiser $\Delta(\mathbf{x})$ revient à minimiser $\sum_{j=1}^{n} a_{n+1,j}(0)\, x_j$. Dans ce cas, la méthode de la poupée russe est une méthode de somme pondérée pour les vecteurs de critères n'appartenant pas à la boîte 0. De plus, si la poupée russe est composée d'un ensemble de simplexes dont les hyperplans obliques ne sont pas parallèles, alors la méthode de la poupée russe se comporte comme une méthode de la somme pondérée avec des poids variables selon la valeur des vecteurs de critères qui sont en dehors de la boîte 0.

Les paramètres des boîtes de qualité de type simplexe peuvent être directement

introduits par le concepteur à l'image des boîtes hyperrectangulaires. Dans ce contexte, l'hyperplan oblique $\mathbf{H}_{n+1}(k)$ est défini au moyen des coefficients $a_{n+1,i}(k) = 1/t_i(k)$, tels que $t_i(k)$ sont des seuils de qualité introduits dans la section précédente.

3.3.5 Définition d'une boîte de qualité au moyen d'un processus d'apprentissage

Une boîte de qualité k est construite par l'intersection d'un hyperrectangle défini par n hyperplans $\mathbf{H}_i(k)$, avec $1 \leq i \leq n$ et un simplexe défini par un hyperplan $\mathbf{H}_{n+1}(k)$. Ainsi, le modèle d'agrégation peut être compensatoire ou non compensatoire dans un voisinage d'un vecteur de critères $\mathbf{x}$. Les relations entre les critères peuvent ensuite être adaptées en fonction du problème de décision multicritère, par un choix adéquat des paramètres des hyperplans. Cependant, il est difficile pour un concepteur d'obtenir ces paramètres à partir des données disponibles, telles que celles fournies par un MOS pour un nombre de critères supérieur à deux. Pour résoudre ce problème, un algorithme de classification automatique, capable de faire face à l'incertitude des données MOS, peut être utilisé. Dans ce contexte, nous utilisons la méthode décrite brièvement ci-dessous.

Globalement, l'algorithme d'apprentissage utilise la méthode de la descente du gradient afin de minimiser la distance entre les hyperplans et des ensembles de points mal classés dans l'espace des critères. Avant d'introduire formellement l'algorithme, les ensembles de points mal classés en dehors de la boîte $\mathbf{B}(k)$ et l'ensemble des points mal classés à l'intérieur de la boîte $\mathbf{B}(k)$, associés à un hyperplan donné $\mathbf{H}_i(k)$, doivent être définis. Soit l une étiquette qui désigne la qualité affectée au point $\mathbf{x}$ dans l'espace des critères. $l = 5 - l_{mos}$, tel que l_{mos} est la qualité affectée au point $\mathbf{x}$ dans l'expérience MOS. Nous rappelons que, lors d'une expérience de type MOS, les utilisateurs attribuent des notes allant de 5 (excellente qualité) à 1 (mauvaise qualité) aux échantillons de la VoIP, en fonction de la qualité perçue.

– **Ensemble des points mal classés en dehors d'une boîte** : $\mathbf{x}$ est dit "mal classé" si $\mathbf{x}$ est en dehors de la boîte $\mathbf{B}(k)$ et $l \leq k$.

  L'ensemble des points mal classés situés en dehors de la boîte $\mathbf{B}(k)$ par rapport

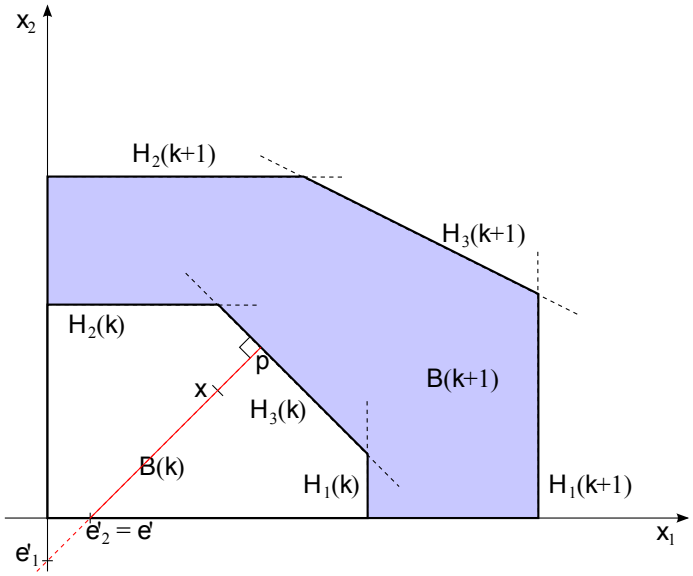


FIGURE 3.6 – $\frac{\|\mathbf{p}-\mathbf{x}\|}{\|\mathbf{p}-\mathbf{e}'\|}$ est la distance normalisée entre le vecteur de critères $\mathbf{x}$ et la boîte $\mathbf{B}(k)$

à l'hyperplan $\mathbf{H}_i(k)$ contient les points mal classés $\mathbf{x}$ dont la distance $\delta_i(\mathbf{x})$ à l'hyperplan $\mathbf{H}_i(k)$ est maximale parmi tous les hyperplans composant la boîte $\mathbf{B}(k)$. L'hyperplan $\mathbf{H}_i(k)$ est appelé "facette active" du point $\mathbf{x}$.

– **Ensemble des points mal classés à l'intérieur d'une boîte** : $\mathbf{x}$ est dit "mal classé" si $\mathbf{x}$ est à l'intérieur de la boîte $\mathbf{B}(k)$ et $l > k$.

L'ensemble des points mal classés à l'intérieur d'une boîte $\mathbf{B}(k)$ par rapport à l'hyperplan $\mathbf{H}_i(k)$ contient les points mal classés $\mathbf{x}$ dont la distance $\delta'_i(\mathbf{x})$ à l'hyperplan $\mathbf{H}_i(k)$ est minimale parmi tous les hyperplans composant la boîte $\mathbf{B}(k)$. $\delta'_i(\mathbf{x})$ est définie d'une façon similaire à la distance $\delta_i(\mathbf{x})$ (équation 3.4).

$$\delta'_i(\mathbf{x}) = \frac{\|\mathbf{p}-\mathbf{x}\|}{\|\mathbf{p}-\mathbf{e}'\|} \tag{3.5}$$

tel que $\mathbf{p}$ est la projection de $\mathbf{x}$ sur $\mathbf{H}_i(k)$ et $\mathbf{e}'_j$ est l'intersection de l'hyperplan $x_j = 0$ avec la normale de $\mathbf{H}_i(k)$ passant par $\mathbf{p}$. $\mathbf{e}' = \mathbf{e}'_j$ en choisissant j tel que $\|\mathbf{p}-\mathbf{e}'\|$ est minimale (voir la figure 3.6). L'hyperplan $\mathbf{H}_i(k)$ est appelé "facette active" de $\mathbf{x}$.

Soit $\mathbf{M}_i(k)$ l'ensemble des vecteurs de critères mal classés associés à l'hyperplan $\mathbf{H}_i(k)$. Cet ensemble est l'union des deux sous-ensembles des vecteurs de critères mal

classés en dehors et à l'intérieur de la boîte $\mathbf{B}(k)$. Le processus d'apprentissage minimise l'erreur quadratique $e(\mathbf{a}_i)$ entre $\mathbf{H}_i(k)$ et les éléments de l'ensemble $\mathbf{M}_i(k)$, en ajustant les coefficients $a_{ij}(k)$ au moyen de la méthode de la descente du gradient.

$$e(\mathbf{a}_i) = \frac{1}{s\mathbf{a}_i^2} \sum_{m=1}^{s} (\mathbf{a}_i \mathbf{x}_m - 1)^2 \tag{3.6}$$

tel que $\mathbf{a}_i$ est le vecteur contenant les coefficients de $\mathbf{H}_i(k)$, $\mathbf{x}_m \in \mathbf{M}_i(k)$ et s est le nombre d'éléments dans $\mathbf{M}_i(k)$. L'équation du gradient est donnée comme suit :

$$\nabla e(\mathbf{a}_i) = \frac{2}{s\left(\mathbf{a}_i^2\right)^2} \sum_{m=1}^{s} (\mathbf{a}_i \mathbf{x}_m - 1) \left(\mathbf{a}_i^2 \mathbf{x}_m - (\mathbf{a}_i \mathbf{x}_m - 1)\mathbf{a}_i\right) \tag{3.7}$$

Particulièrement, pour un hyperplan $\mathbf{H}_i(k)$ de l'hyper-rectangle composant la boîte, cette équation devient :

$$\frac{\partial e}{\partial a_{ii}} = \frac{2}{a_{ii}^2} \left(\frac{\sum_{m=1}^{s} x_{mi}}{s} - \frac{1}{a_{ii}} \right) \tag{3.8}$$

Le gradient est nul pour :

$$a_{ii} = \frac{s}{\sum_{m=1}^{s} x_{mi}} \tag{3.9}$$

L'algorithme d'apprentissage est composé d'une boucle principale, qui optimise d'une façon itérative les paramètres d'une boîte. Dans ce contexte, les coefficients des différents hyperplans composant l'hyperrectangle sont mis à jour au moyen de la fonction "HyperRectangleUpdate", tandis que ceux de l'hyperplan oblique sont mis à jour au moyen de la fonction "ObliqueUpdate". La fonction "isMisclassified" retourne "vrai" si un vecteur de critère $\mathbf{x}$ associé à une qualité donnée l est mal classé par rapport à la boîte de qualité k. Par ailleurs, la fonction "GetQuadraticCost" est utilisée pour calculer la somme des erreurs quadratiques données par l'équation (3.6) pour tout hyperplan d'une boîte de qualité donnée. Enfin, le processus d'apprentissage se termine quand la variation de l'erreur quadratique est inférieure à un seuil déterminé ε. L'algorithme utilise les notations suivantes :

- $\mathbf{T}$: est l'ensemble d'apprentissage composé par des couples $(\mathbf{x}, l)$ où l est un indicateur de qualité subjective associée au vecteur de critères $\mathbf{x}$. $\mathbf{T}$ peut être le

résultat d'une expérience de type MOS.

– α : est le pas utilisé dans la méthode de la descente du gradient. α doit être positif et suffisamment petit pour garantir la convergence de l'algorithme.

L'algorithme d'apprentissage présenté ci-dessous doit être appliqué à chaque boîte de la poupée russe, en commençant par la plus grande boîte jusqu'à la plus petite (boîte 0). Ainsi, la distance δ_i est toujours correctement définie.

Algorithme 2: Algorithme d'apprentissage

Données :
$\mathbf{T}$: l'ensemble d'apprentissage,
k : index d'une boîte,
n : nombre de critères dans le système,
α : pas de la méthode du gradient,
ε : critère d'arrêt de l'algorithme.
Résultat :
$\mathbf{a}$: matrice (a_{ij}) où a_{ij} est le j-ième coefficient de $\mathbf{H}_i(k)$

$\mathbf{a} \leftarrow$ RandomInit(n)
$e \leftarrow 0$
répéter
  HyperRectangleUpdate($\mathbf{a}$, n, k, $\mathbf{T}$)
  ObliqueUpdate($\mathbf{a}$, n, k, $\mathbf{T}$, α)
  $p \leftarrow e$
  $e \leftarrow$ GetQuadraticCost($\mathbf{a}$, n, k, $\mathbf{T}$)
jusqu'à $|e - p| < \varepsilon$;

Procédure HyperRectangleUpdate($\mathbf{a}$, n, k, $\mathbf{T}$)

pour chaque $i \in [1, n]$ **faire**
  $s_i \leftarrow 0$
  $c_i \leftarrow 0$
pour chaque $(\mathbf{x}, l) \in \mathbf{T}$ **faire**
  si *isMisclassified*$((\mathbf{x}, l), \mathbf{a}, k)$ **alors**
    $i \leftarrow$ GetActiveHyperplaneIndex$(\mathbf{x}, \mathbf{a})$
    si $i \leq n$ **alors**
      $s_i \leftarrow s_i + 1$
      $c_i \leftarrow c_i + x_i$ // x_i est la i-ème coordonnée de $\mathbf{x}$
pour chaque $i \in [1, n]$ **faire**
  si $s_i > 0$ **alors**
    $a_{ii} \leftarrow s_i / c_i$ // voir l'équation (3.9)

Procédure ObliqueUpdate($\mathbf{a}$, n, k, $\mathbf{T}$, α)

$s \leftarrow 0$
$\mathbf{g} \leftarrow 0$
pour chaque $(\mathbf{x}, l) \in \mathbf{T}$ **faire**
  si *isMisclassified*$((\mathbf{x}, l), \mathbf{a}, k)$ **alors**
    $i \leftarrow$ GetActiveHyperplaneIndex$(\mathbf{x}, \mathbf{a})$
    si $i = n + 1$ **alors**
      $s \leftarrow s + 1$
      $\mathbf{g} \leftarrow \mathbf{g} + (\mathbf{a}_{n+1}\mathbf{x} - 1)\left(\mathbf{a}_{n+1}^2\mathbf{x} - (\mathbf{a}_{n+1}\mathbf{x} - 1)\mathbf{a}_{n+1}\right)$
      // voir l'équation (3.7)
$\mathbf{a}_{n+1} \leftarrow \mathbf{a}_{n+1} - \alpha\mathbf{g}/s$

3.4 Conclusion

La plupart des méthodes de décision multicritère classiques utilisent des paramètres qui sont déduits des préférences du décideur. Ces paramètres sont souvent des poids qui quantifient l'importance de chaque critère dans le processus de décision multicritère. Dans ce chapitre, nous nous sommes intéressés au cas de la prise de décision multicritère dans les systèmes autonomes et dynamiques dont le front de Pareto varie en fonction du temps. Bien évidemment, il est impossible de prévoir une interaction avec le décideur dans ce type de systèmes.

Dans ce contexte, nous avons présenté une méthode de décision multicritère qui s'inspire du principe d'empilement d'une poupée russe pour construire une fonction d'utilité dans l'espace des critères. Cette fonction d'utilité utilise des paramètres déterminés lors de la phase de conception. En effet, les paramètres de la méthode sont soit directement déduits des normes et spécifications techniques des systèmes, soit éventuellement obtenus par l'intermédiaire d'une méthode de classification automatique exécutée sur un MOS. Plus précisément, la méthode de la poupée russe permet trois types d'adaptation distincts :

- Elle peut adapter l'importance des critères selon leurs valeurs, grâce aux multiples boîtes de qualité englobantes,
- Elle offre la possibilité de prendre en compte les différentes relations qui existent entre critères, en l'occurrence la compensation (boîtes de type simplexe), la non-compensation (boîtes de type hyperrectangulaire) et une combinaison de ces

dernières (intersection d'un simplexe et d'un hyperrectangle),

– Elle offre la possibilité d'apprendre ses paramètres à partir de données dont l'acquisition peut être automatisée, ce qui lui permet d'adapter sa forme si cela est utile à la maximisation des performances du système.

Dans le chapitre suivant, nous allons évaluer les performances de la méthode dans un cas de prise de décision multicritère, pour un système autonome et dynamique, en l'occurrence le routage multicritère dans les réseaux ad hoc sans fil.

Chapitre 4

Application au routage multicritère dans les réseaux ad hoc sans fil

4.1 Introduction

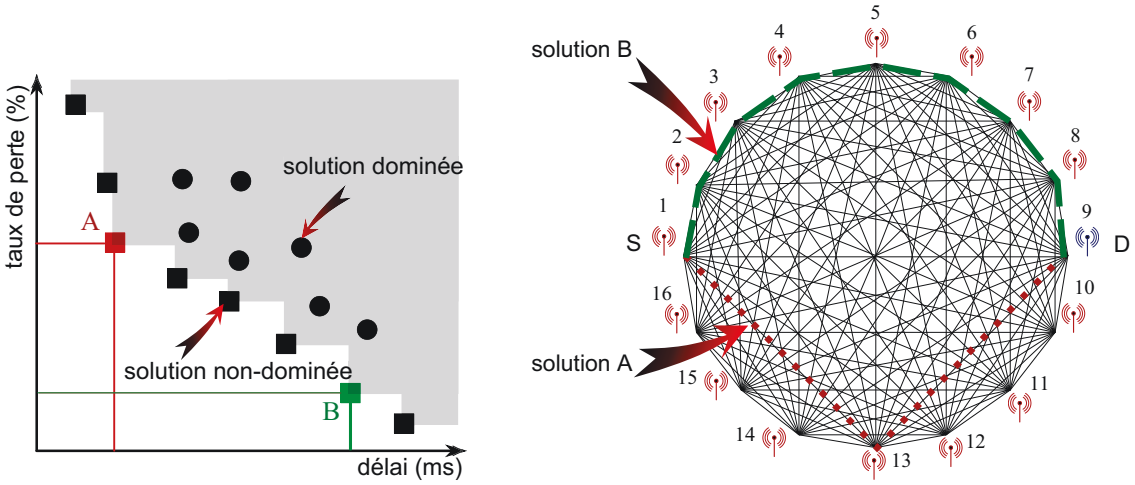
Les réseaux ad hoc sans fil sont des réseaux auto-adaptatifs formés par un ensemble de terminaux sans fil qui établissent une topologie afin de communiquer. L'objectif de la tâche du routage dans ces réseaux est de trouver un chemin efficace (au sens large) entre un nœud source donné et un nœud destinataire. Comme la portée des nœuds est relativement petite, les chemins utilisés pour la communication sont souvent composés de plusieurs nœuds intermédiaires. De plus, la tâche du routage doit optimiser un certain nombre de critères relatifs à la qualité de service QoS (*Quality of Service*), en l'occurrence le délai, le taux de perte, la gigue, etc. Dans ce contexte, la majorité des protocoles de routage introduits dans la littérature prennent uniquement le nombre de sauts comme métrique à optimiser et négligent, par conséquent, les exigences des applications en termes de différents critères de QoS. Cependant, le problème de routage dans les réseaux ad hoc sans fil est typiquement une problématique de décision multicritère, dans ce sens que plusieurs critères, souvent contradictoires, sont impliqués dans le processus de décision. Dans ce chapitre, nous comparerons la méthode de la poupée russe à la méthode de la somme pondérée, dans le cas du routage dans les réseaux sans fil. Ce problème consiste à choisir le meilleur chemin en fonction des exigences

des applications du réseau. Par exemple, une application de type VoIP peut accepter des taux de pertes relativement élevés, contrairement au délai qui doit être petit. D'un autre côté, une application de type transfert de fichier tel que FTP (*File Transfer Protocol*) est sensible au critère du taux de perte, par contre, elle peut accepter des délais relativement élevés. La figure 4.1 illustre schématiquement la problématique de routage multicritère dans les réseaux ad hoc sans fil. La partie droite de la figure illustre la topologie du réseau qui comporte plusieurs chemins entre le nœud source S et le nœud destinataire D. Chaque chemin est caractérisé par ses propres métriques (ou critères) tels que le délai, le taux de perte, la gigue, etc. Quant à la figure de gauche, celle-ci représente les alternatives non-dominées au niveau de la source. Chaque point dans l'espace des critères représente un chemin distinct dans le réseau. Nous pouvons remarquer qu'une problématique existe en ce qui concerne les chemins appartenant au front de Pareto. En effet, le chemin A est bon en termes de délai, mais relativement mauvais en termes de taux de perte. L'inverse est vrai pour le chemin B. De ce fait, le protocole de routage doit s'appuyer sur une méthode de décision multicritère, qui devra gérer le processus du choix des chemins en fonction des exigences des applications routées.

Ce chapitre est organisé comme suit. Premièrement nous exposons, dans la section 4.2, les travaux de recherche qui ont tenté d'incorporer la notion de décision multicritère dans le routage. Ensuite, la section 4.3 donne un aperçu du principe de l'apprentissage par renforcement qui est utilisé dans le protocole de routage. Puis, nous introduisons, dans la section 4.4, le protocole de routage, qui utilise la méthode de la poupée russe, afin de trouver les chemins les plus adaptés aux différents applications, selon leurs exigences. Ensuite, la section 4.5 montre une évaluation des performances de la méthode de la poupée russe comparée à la méthode de la somme pondérée. Enfin, nous concluons ce chapitre dans la section 4.6.

4.2 État de l'art

La majorité des protocoles de routage introduits dans la littérature utilisent les poids de quantification afin de marquer l'importance des critères de QoS dans le pro-



a. Solutions dominées et non-dominées b. La topologie du réseau

FIGURE 4.1 – Problématique de routage multicritère dans un réseau sans fil

cessus de routage. Une fois que l'importance des critères est quantifiée, une somme pondérée des critères est utilisée afin de transformer le problème de routage multicritère en un problème de routage monocritère. Cependant, dans la majorité des études, aucune indication n'est donnée sur la méthode utilisée pour choisir la valeur des poids de quantification. Dans cette section, nous passerons en revue les études pertinentes qui ont tenté d'apporter des solutions au problème de routage dans les réseaux ad hoc sans fil.

Dans [ED06], les auteurs minimisent une somme pondérée de deux critères, en l'occurrence le délai et la distance entre la source et la destination, afin d'optimiser le processus de routage dans un réseau ad hoc. Les auteurs fixent les valeurs des poids pondérant les deux critères moyennant plusieurs scénarios de simulation. Similairement, dans [MTT+06], les auteurs utilisent la méthode de la somme pondérée pour optimiser trois critères, notamment le délai, l'énergie et le taux de perte binaire BER (*Bit Error Rate*) dans un réseau sans fil. Le choix des valeurs des poids utilisés dans cette étude est laissé libre à l'utilisateur. Dans [GDRMB09], les auteurs formalisent le problème de routage comme un problème de décision multicritère et proposent de le résoudre moyennant une somme pondérée. Cependant, aucune méthode n'est proposée pour fixer les valeurs des poids relatifs aux critères. Par contre, les auteurs ont présenté une étude de sensibilité du processus de décision multicritère par rapport aux

valeurs des poids. Les auteurs concluent par la remarque de l'importance de l'estimation des valeurs des poids et observent que ces derniers dépendent considérablement des exigences des applications routées. Dans [SK06], les auteurs traitent le problème du choix de la meilleure connexion réseau comme un problème de décision multicritère. Le but est d'optimiser deux critères contradictoires, en l'occurrence le coût (prix) et le temps de transmission, quand un ensemble d'applications (données, vidéos, voix, etc) sont transférées via plusieurs réseaux de communication. Le problème est résolu par la minimisation de la distance de Chebyshev à un point de référence. Comme dans les études présentées précédemment, les auteurs affirment que les poids utilisés dans l'expression de la distance doivent être introduits par le décideur en personne. Cependant, les auteurs défendent leur travail, en stipulant que le choix de la solution finale est peu dépendant des valeurs des poids, contrairement au choix de la solution de référence. Dans [REM08], les auteurs ont utilisé la distance de *Tchebychev* au vecteur de critère idéal pour sélectionner un chemin parmi l'ensemble des chemins non-dominés qui existent entre une source et une destination dans un réseau de données. L'ensemble des chemins non-dominés est obtenu au moyen d'un algorithme dédié [GBR06, Mar84], quant au vecteur idéal, il est obtenu en prenant la valeur optimale de chaque critère parmi l'ensemble des chemins non-dominés. Bien que cette méthode permet de résoudre la problématique de prise de décision multicritère en temps réel, elle souffre néanmoins des inconvénients relatifs à la méthode de la distance à un objectif de référence, notamment le choix des valeurs des poids de pondération utilisés dans l'expression de la distance de *Tchebychev* (voir la section 2.4.3).

Dans [RD04], les auteurs ont tenté de résoudre le problème du routage *multicast* en utilisant les algorithmes génétiques multiobjectifs (MOGA) [FF$^+$93]. Le but de ce travail est de construire un arbre de routage *multicast* qui optimise deux critères en l'occurrence le délai et la bande passante. Le choix du meilleur arbre *multicast* est effectué par le décideur à la fin de l'optimisation (quand l'algorithme génétique trouve les solutions non-dominées) en fonction des exigences des applications. Dans [Ish06], les auteurs utilisent une solution de référence fournie par le décideur, afin de guider le processus de recherche lors de l'étape de sélection de l'algorithme génétique

multiobjectif (MOGA). Cependant, un choix adéquat de la solution de référence est exigé, car la méthode en dépend grandement.

4.3 Aperçu sur l'apprentissage par renforcement

L'apprentissage par renforcement [Sut98] est une branche de l'intelligence artificielle qui permet de reproduire le mécanisme d'apprentissage des animaux par des ordinateurs. En effet, un animal apprend à connaître son environnement au moyen du mécanisme action/récompense, qui est aussi le fondement de l'apprentissage par renforcement. La figure 4.2 illustre le schéma général de fonctionnement d'un système utilisant l'apprentissage par renforcement. Dans ce contexte, le système est représenté par un agent composé de plusieurs états, qui ne sont pas nécessairement connus à l'avance. L'agent choisit les meilleures actions qui maximisent l'espérance de ses gains et se déplace d'un état à un autre, en fonction de la réponse de l'environnement. Mathématiquement parlant, soit $\mathcal{S}$ l'ensemble des états possibles dans le système. A chaque instant t, l'agent perçoit un état $s_t \in \mathcal{S}$ de son environnement et sélectionne une action $a_t \in \mathcal{A}(s_t)$, où $\mathcal{A}(s_t)$ est l'ensemble des actions possibles à l'état s_t. En retour, l'agent reçoit de son environnement une récompense immédiate $r_{t+1} \in \mathcal{R}$, où $\mathcal{R}$ est l'ensemble des récompenses, puis se déplace à l'état s_{t+1} (voir la figure 4.2). A chaque instant t, l'agent suit une stratégie (ou politique) donnée, notée $\pi_t(s, a)$, qui est la probabilité que l'agent choisisse l'action a_t sachant que ce dernier est dans l'état s_t. Les méthodes d'apprentissage par renforcement adaptent la politique de l'agent en fonction de son expérience dans le temps. Pour ce faire, les méthodes d'apprentissage par renforcement définissent une fonction de valeur (*value function*) qui tente de prévoir l'espérance des gains futurs que l'agent peut recevoir s'il se trouve dans un certain état, ou s'il exécute une certaine action en se trouvant dans un certain état. Plus précisément, la fonction de valeur $V^{\pi}(s)$ d'un état s suivant la politique π est définie, par exemple, comme suit :

$$V^{\pi}(s) = E_{\pi}\left\{R_t / s_t = s\right\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} / s_t = s\right\}$$

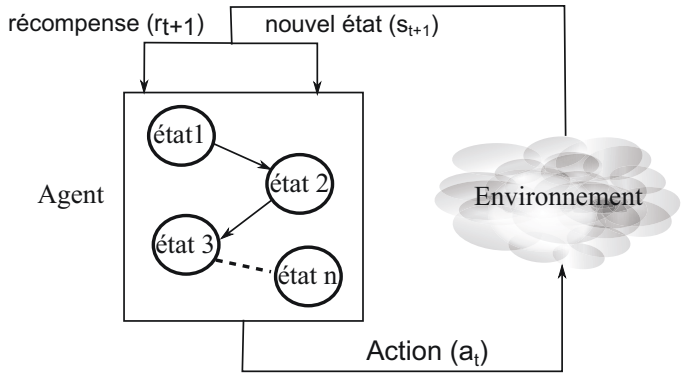


FIGURE 4.2 – Principe de l'apprentissage par renforcement

où $E_\pi\{.\}$ est l'espérance mathématique. $\gamma \in [0,1]$ est un nombre réel qui quantifie l'importance des récompenses futures dans le processus d'apprentissage par renforcement : une valeur de γ proche de 1 signifie que les récompenses futures sont importantes dans le calcul de la fonction de valeur. $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$.

De même, la fonction de valeur d'un couple état/action (*action-value function*) $Q^\pi(s,a)$ suivant la politique π est définie comme suit :

$$Q^\pi(s,a) = E_\pi\{R_t/s_t = s, a_t = a\} = E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}/s_t = s, a_t = a\right\}$$

Les valeurs $V^\pi(s)$ et $Q^\pi(s,a)$ peuvent être estimées en calculant leur moyennes empiriques au cours du temps. Pour ce faire, il suffit que l'agent suive la politique π puis calcule progressivement la moyenne empirique en fonction des récompenses qu'il reçoit de son environnement. La moyenne convergera dans le temps vers la valeur $V^\pi(s)$, respectivement $Q^\pi(s,a)$, si le nombre d'occurrences de l'état s, respectivement couple état/action (s,a) est assez grand. Notons que ce procédé d'estimation de la valeur de $V^\pi(s)$ et $Q^\pi(s,a)$ est appelé la méthode de Monte Carlo. Il existe d'autres méthodes qui permettent aussi d'estimer les valeurs des fonctions de valuation, notamment les méthodes dites *TD-learning* (*Temporal-Difference Learning*) ou encore *Q-Learning* [Sut98].

Le but de l'agent est de constituer une politique optimale π^* qui permet de maximiser ses gains. En d'autres termes, l'agent aspire à constituer une politique π^* dont la fonction de valeur $V^*(s)$ est définie comme suit :

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

La politique π^* doit aussi maximiser la fonction de valeur état-action comme suit :

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

Finalement, nous pouvons formaliser l'espérance des gains que percevra un agent se trouvant dans l'état s et exécutant l'action a, suivant la politique optimale π^*, comme suit :

$$Q^*(s,a) = E\left\{r_{t+1} + \gamma V^*(s_{t+1}) / s_t = s, a_t = a\right\} \tag{4.1}$$

L'expression 4.1 découle directement du principe d'optimalité de *Bellman* [Sut98].

4.3.1 Application au problème de routage multicritère

Dans le cas de la problématique de routage, tous les nœuds du réseau sont considérés comme des agents. De plus, pour chaque application dans le réseau, tel que la VoIP, la vidéo ou le transfert de fichier, il existe une politique de routage π spécifique qui satisfasse plus au moins ses exigences en termes de QoS. Notons que le problème de recherche de l'ensemble des routes multicritères optimales (front de Pareto) est un problème NP-difficile [WC96]. Par conséquent, notre approche de routage fondée sur le principe d'apprentissage par renforcement est une heuristique qui aspire à approcher le front de Pareto du problème en un temps raisonnable.

Soient a, s, n, d la classe de l'application, le nœud source, un nœud voisin et le nœud destination, respectivement. Un nœud r doit choisir un nœud voisin n^* en fonction de l'application a et la destination d de telle façon que le vecteur de critères au niveau de la destination satisfasse les exigences multicritères de l'application a. Les vecteurs de critères sont considérés comme des réalisations d'un vecteur aléatoire $\mathbf{c}$. Chaque nœud du réseau possède une table de routage (fonction de valeur) dans laquelle il mémorise l'estimation des espérances des vecteurs de critères $\mathbf{Q}(a,d,r,n) = E\left\{\mathbf{c}/a,d,r,n\right\}$. Les paramètres a et d sont contenus dans le paquet d'apprentissage introduit dans la sous-

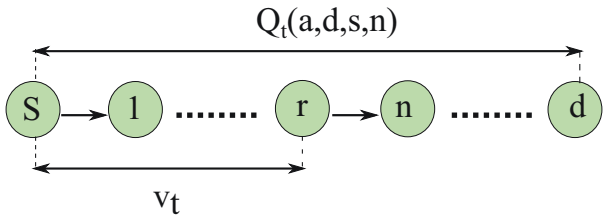


FIGURE 4.3 – Illustration d'un chemin entre la source et la destination

section suivante. L'espérance des récompenses est mise à jour au niveau de chaque source du réseau comme suit :

$$\begin{cases} \mathbf{Q}_t(a,d,s,n) = (1-\gamma)\,\mathbf{Q}_{t-1}(a,d,s,n) + \gamma \mathbf{c}_t & \text{si } t > 0 \\ \mathbf{Q}_0(a,d,s,n) = 0 & \text{sinon} \end{cases} \tag{4.2}$$

où $\mathbf{c}_t$ est la récompense immédiate calculée au niveau de la destination pour le paquet numéro t. Les coordonnées de $\mathbf{c}_t$ sont les différents critères du problème traités, en l'occurrence les critères de qualité de service pour l'application de routage.

Finalement, le meilleur voisin n^* pour le nœud r est le voisin qui maximise l'espérance de la qualité de service multicritère $\mathbf{Q}(a,d,r,n)$. Le meilleur voisin n^* est choisi par la méthode de décision multicritère utilisée comme suit :

$$n^* = \underset{n \in \mathbf{N}(r)}{\arg\max}\, u\left(\mathbf{Q}_t(a,d,r,n) + \mathbf{v}_t\right) \tag{4.3}$$

Où,

- $u(\mathbf{c})$ est une fonction d'utilité ;
  - $u(\mathbf{c}) = m - \Delta(\mathbf{c})$ pour la méthode de la poupée russe, où m est l'index de la plus grande boîte de la poupée russe et $\Delta(\mathbf{c})$ est la distance au système de boîtes englobantes ;
  - $u(\mathbf{c}) = 1 - \sum_{i=1}^{n} w_i c_i$ pour la méthode de la somme pondérée, tel que $\sum_{i=1}^{n} w_i = 1$ et c_i est la valeur du critère i qui est normalisé de telle sorte que $c_i \in [0,1]$;
- $\mathbf{N}(r)$ est la liste des voisins directs du nœud r ;
- $\mathbf{v}_t$ est le vecteur des critères relatif au chemin (suivi par le paquet t) entre le nœud source s et le nœud courant r (voir la figure 4.3). $\mathbf{v}_t = 0$ au niveau de la source s.

4.4 Protocole de routage multicritère utilisant l'apprentissage par renforcement

Les méthodes classiques de routage dans les réseaux ad hoc sans fil ne sont pas multicritères, ce qui justifie la conception d'un nouveau protocole de routage. Dans cette section, nous présentons le protocole RLRP (*Reinforcement Learning Routing protocol*) qui est un protocole de routage multicritère fondé sur le principe de l'apprentissage par renforcement.

4.4.1 Aperçu général du fonctionnement du protocole RLRP

Le protocole RLRP opère en deux phases principales, en l'occurrence la phase de découverte et maintenance de routes et la phase de routage des paquets de données. Dans la phase de découverte et maintenances de routes, chaque source (un nœud donné s dans le réseau qui envoie des paquets à une destination d) envoie des paquets spéciaux, appelés "paquets d'apprentissage", afin de découvrir une ou plusieurs routes vers la destination. Pour ce faire, le nœud source s choisit aléatoirement un voisin de sa liste des voisins directs et lui envoie un paquet d'apprentissage appelé, *Forward Learning Packet* (FLP) qui est destiné à la destination d. Le voisin direct du nœud s qui reçoit le paquet FLP examine sa table de routage, pour savoir s'il existe ou pas un chemin vers la destination d. Dans le cas où un chemin existe, le voisin direct de la source s envoie le paquet FLP au prochain nœud sélectionné par la méthode de décision multicritère. Dans le cas contraire, le voisin direct de la source sélectionne aléatoirement un nœud dans sa liste de voisins directs et lui envoie à son tour le paquet FLP. Ainsi, de proche en proche, le paquet FLP transite entre les nœuds intermédiaires, jusqu'à ce qu'il arrive au niveau de la destination d. Cependant, le paquet FLP peut être supprimé au cours du temps, s'il passe par un nœud donné pour la deuxième fois (cas d'une boucle) ou si le nombre de nœuds visités dépasse un seuil prédéfini. Notons que, pendant son passage de la source vers la destination le paquet FLP est mis à jour de plusieurs façons. Premièrement, la liste des nœuds déjà visités est stockée dans un champ spécial appelé "mémoire". Enfin, les différents critères, tels que le nombre

de sauts, le délai et le taux de perte, sont estimés au fur et à mesure que le paquet FLP s'approche de la destination. Au niveau de la destination, le paquet FLP est transformé en un paquet d'apprentissage, appelé *Backward Learning Packet* (BLP) qui a comme source la destination d et comme destination la source s. Le paquet BLP suit le chemin inverse emprunté par le paquet FLP, en parcourant la liste des nœuds stockés dans la mémoire du paquet FLP, de la fin jusqu'au début, jusqu'à ce qu'il atteigne la source s. Notons qu'à l'image d'un paquet FLP, un paquet BLP peut être perdu en route, à cause des problèmes de transmission. Dans le cas où le paquet BLP atteint la source, cette dernière l'utilise pour mettre à jour sa table de routage au moyen de l'expression 4.2. Si le paquet est perdu, la source pénalise dans sa table de routage le voisin responsable. Ainsi, de proche en proche, les nœuds découvrent des chemins divers vers la destination. Ces chemins optimiseront différemment les critères de qualité de service, ce qui nous amène à une problématique de décision multicritère des chemins en fonction des exigences des différentes applications.

4.4.2 Présentation détaillée

Vue statique du protocole RLRP

La figure 4.4 illustre les différentes composantes qui forment le protocole RLRP. Comme nous pouvons le constater sur le diagramme, le protocole RLRP comprend une classe principale, appelée RLRP, qui est instanciée dans tous les nœuds du réseau. Cette classe porte l'identificateur du nœud en question. La classe RLRP englobe plusieurs autres classes qui représentent l'aspect statique du système :

- Voisin : un nœud voisin est identifié par son identificateur et l'instant auquel il est rentré en contact avec le nœud courant.
- Table de routage : chaque nœud du réseau maintient une table de routage pour chaque application. Chaque application utilise à son tour sa propre méthode de décision multicritère. La table de routage maintient aussi la liste des routes découvertes par l'envoi des paquets d'apprentissage. Chaque route est indexée par son premier nœud, qui est à son tour un voisin direct du nœud courant.
- Paquet : la classe Paquet généralise les trois formes de paquets utilisés par le

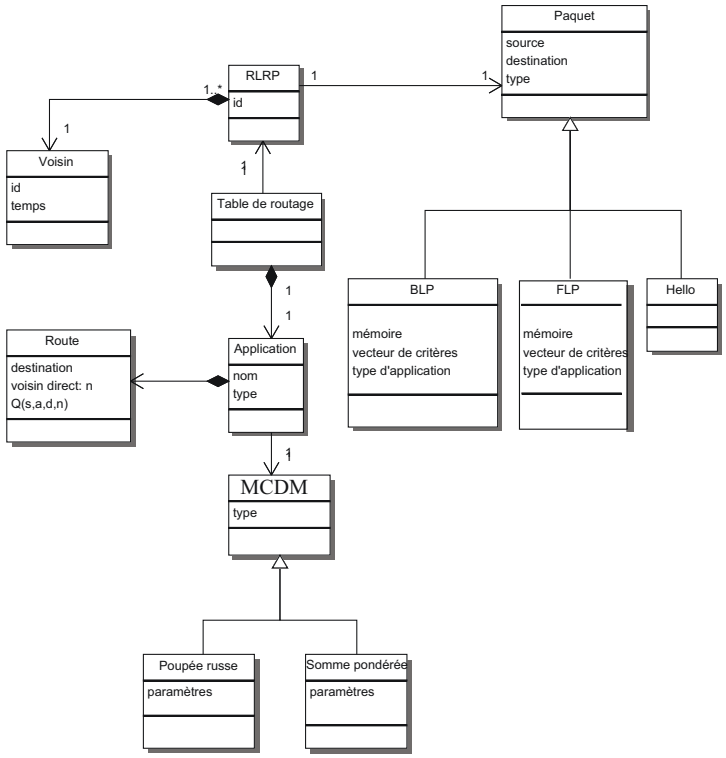


FIGURE 4.4 – Diagramme de classe du protocole RLRP

protocole RLRP, en l'occurrence les paquets FLP, BLP et "Hello". Le paquet "Hello" est employé afin de découvrir la liste des voisins directs du nœud courant. La classe Paquet encapsule les attributs source, destination et le type de paquet. Les paquets d'apprentissage FLP et BLP encapsulent les propriétés importantes suivantes : type d'application, liste des nœuds intermédiaires déjà visités et le vecteur de critères.

Vue dynamique du protocole RLRP

Dans cette sous-section, nous présentons l'aspect dynamique du protocole RLRP notamment la mécanique qui permet de découvrir la topologie du réseau et le routage des paquets de données. En plus des différentes structures de données présentées ci-dessus, le protocole RLRP utilise des *timers* afin de réguler le mécanisme d'apprentissage et de découverte de voisins directs. La figure 4.5 illustre le fonctionnement de la classe principale du protocole RLRP. Cette dernière simule le comportement d'un nœud du réseau en fonction des événements qu'il reçoit de son environnement. D'abord,

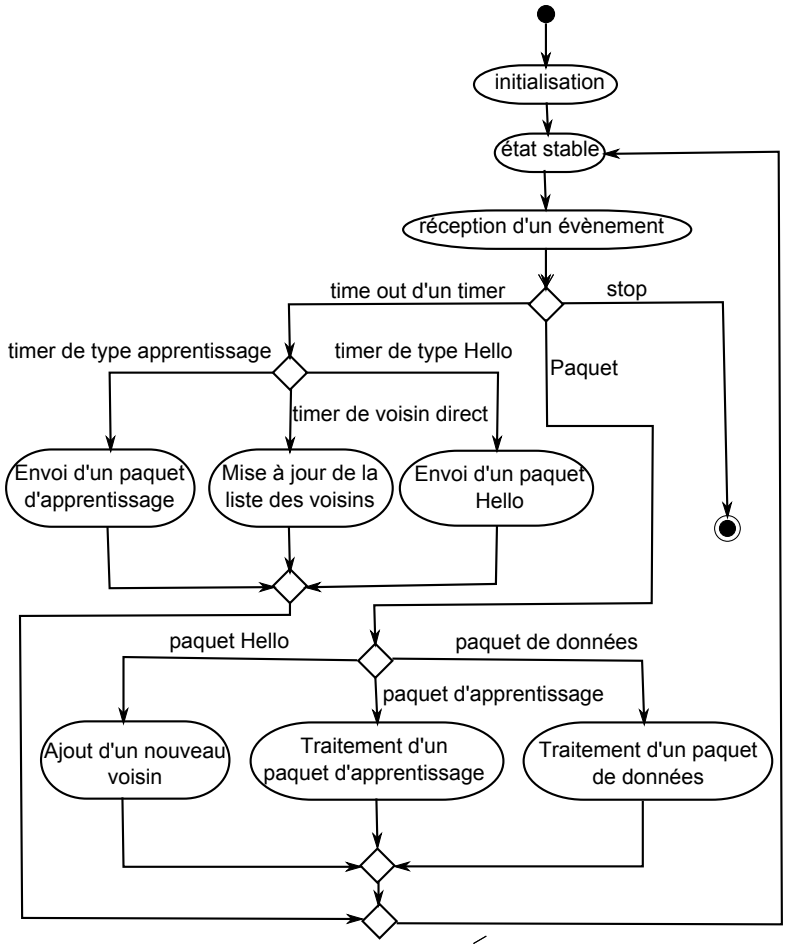


FIGURE 4.5 – Diagramme d'activité de la classe RLRP

le protocole commence par une étape d'initialisation où des valeurs sont attribuées à tous les paramètres notamment les valeurs des différents *timers*, à savoir le *timer* d'envoi de message hello et le *timer* de la mise à jour de la liste des voisins. Ensuite, le protocole passe à l'étape de stabilité, où le protocole est prêt à recevoir tous les événements de son environnement. Finalement, chaque réception d'un événement déclenche un processus réactif qui contribue à la mise à jour des différentes structures de données du protocole RLRP, en l'occurrence la table de routage et la liste des voisins directs.

- Envoi d'un paquet "Hello" : le nœud courant crée un paquet "Hello", initialise le champ source avec son identificateur, le champ destination par l'adresse de *broadcast* puis envoie le paquet à travers l'interface de la couche réseau.
- Envoi d'un paquet d'apprentissage : d'abord, le nœud courant sélectionne une

application et un voisin direct aléatoirement. Puis, il initialise les champs source, destination avec son identificateur et la destination courante et envoie le paquet à travers l'interface entre les couches réseau et liaison de données.

- Mise à jour de la liste des voisins : le nœud courant examine la liste des voisins directs. Les voisins qui n'ont pas envoyé de message "Hello" depuis une durée de temps déterminée (typiquement, trois fois l'intervalle de temps nécessaire à l'envoi d'un message "Hello") seront supprimés de la liste des voisins directs.
- Ajout d'un nouveau voisin : à la réception d'un paquet "hello", le nœud courant examine sa liste des voisins directs. Si le voisin n'existe pas déjà dans la liste, alors le nouveau voisin y est inséré. Dans le cas contraire, la date courante de la réception d'un message "Hello" pour le voisin courant est mise à jour.
- Traitement d'un paquet d'apprentissage : premièrement, le nœud courant examine le champ type de paquet d'apprentissage reçu, qui peut être soit un paquet FLP ou BLP. Dans le cas où le paquet reçu est un paquet FLP, le nœud courant examine le champ destination. Si le paquet lui est destiné, alors le nœud courant crée un nouveau paquet de type BLP et le renvoie à la source du paquet FLP. Dans le cas où le paquet reçu est un paquet FLP, le nœud courant l'oriente à un voisin direct, s'il existe un chemin vers la destination au niveau du nœud courant, ou l'oriente vers un voisin sélectionné aléatoirement dans le cas contraire.
- Traitement d'un paquet de données : à la réception d'un paquet de données, le nœud courant examine sa destination. Dans le cas où la destination finale du paquet de données est le nœud courant, le paquet est envoyé à la couche transport. Dans le cas contraire, le nœud courant examine sa table de routage pour savoir s'il existe un chemin vers la destination ou pas. Dans le cas où un chemin existe, le paquet de données est orienté vers le voisin direct indexant le chemin trouvé ; dans le cas contraire, le paquet de données est supprimé.

4.5 Expérimentations

Dans cette section, nous évaluons les performances de la méthode de la poupée russe combinée avec le protocole de routage RLRP dans un cas concret de réseau ad

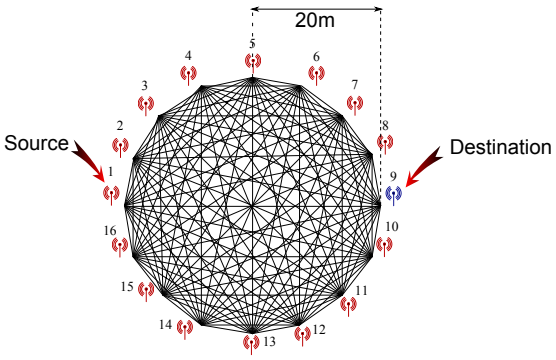


FIGURE 4.6 – Scénario de simulation

hoc. Pour ce faire, nous avons implanté le protocole RLRP, la méthode de la poupée russe et la méthode de la somme pondérée dans le simulateur ns-2 (*network simulator 2*) [BEF+00]. Le simulateur ns-2 est un simulateur à événements discrets qui offre une large gamme de fonctionnalités qui permettent de modéliser l'état réel de fonctionnement de plusieurs réseaux, notamment les réseaux ad hoc. Le simulateur ns-2 est très populaire dans le domaine de la simulation des réseaux ; en témoignent le grand nombre d'articles scientifiques qui l'utilisent dans l'étape d'évaluation. Dans les sections suivantes, nous présentons les expérimentations réalisées sur trois types de scénarios :

- un problème "jouet" utilisant un ensemble d'alternatives non-dominées prédéfinies,
- un cas de routage dans lequel le modèle *shadowing* est utilisé comme modèle de propagation,
- un cas de routage où le taux de perte est une fonction quadratique de la distance entre l'émetteur et le récepteur.

4.5.1 Scénario

Le réseau est formé de 16 nœuds répartis uniformément sur un anneau de rayon 20 m (Voir la figure 4.6). Ce scénario peut être vu comme un ensemble de points d'accès qui forment un réseau *mesh*. Les nœuds 1 et 9 forment un couple source et destination. Tous les nœuds du réseau, excepté le nœud destination, envoient des pa-

quets d'apprentissage pour explorer la topologie et découvrir des routes multicritère vers la destination. En plus des paquets d'apprentissage, le nœud 1 envoie des paquets CBR (*Constant Bit Rate*) simulant une application VoIP avec un débit de $16Ko/s$. Au niveau de la couche MAC (*Medium Access Control*), nous utilisons le standard IEEE 802.11 avec un débit de transmission de l'ordre de 2 Mbps. Dans ces simulations, nous observons deux critères, en l'occurrence le délai de bout en bout (D), et le taux de perte de bout en bout (L). Le délai de bout en bout d'un paquet donné est la somme des délais individuels (d_i) engendrés par chaque nœud intermédiaire i qui compose le chemin entre la source et la destination (voir la figure 4.7). Les délais individuels d_i sont engendrés par les congestions qui apparaissent au niveau des nœuds intermédiaires pendant la communication. Une congestion se produit quand un nœud donné est surpassé par la réception, transmission et relai des différents paquets appartenant à de multiples applications qui transitent dans le réseau. Le délai de bout en bout est donné comme suit :

$$D = \sum_{i=1}^{n-1} d_i \tag{4.4}$$

où $d_i, i \in [1, n-1]$ est un nombre réel positif généré par une loi logistique de moyenne $100ms$ et d'écart type $80ms$ [AEES08].

D'autre part, le critère du taux de perte de bout en bout est fonction des taux de pertes l_i des liens composant le chemin emprunté par les paquets allant de la source vers la destination. l_i est provoqué par l'atténuation du signal en fonction de la distance entre les nœuds successifs i et $i+1$, ainsi que les interférences générées par le voisinage du nœud récepteur. Le taux de perte de bout en bout L est donné comme suit :

$$L = 1 - \prod_{i=1}^{n-1} (1 - l_i) \tag{4.5}$$

l_i sont générés de deux manières, en fonction du scénario testé : premièrement, en utilisant le modèle de propagation *Shadowing* [SAZ07] avec le paramètre "*path loss exponent*" égal à 3 et un écart type égal à 4. Deuxièmement, $l_i = \frac{d^2}{40000}$ pour le scénario incluant un modèle de perte quadratique, avec d exprimée en mètres.

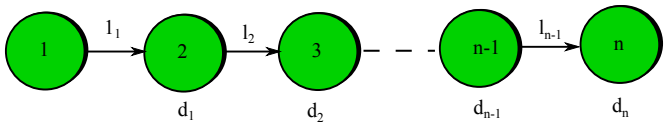


FIGURE 4.7 – Illustration d'une route entre la source et la destination

4.5.2 Structure de la poupée russe

La structure de la poupée russe qui représente les exigences de l'application VoIP est montrée dans la figure 4.8. Les formes des boîtes ont été apprises au moyen de la méthode de classification automatique introduite dans le chapitre 3. L'apprentissage a été effectué sur une expérience MOS simulée contenant 1000 couples $(\mathbf{c_i}, q_i)$, tels que $\mathbf{c_i}$ sont des vecteurs de critères mesurés au niveau de la destination et q_i est une valeur de qualité subjective. Ces couples $(\mathbf{c_i}, q_i)$ de la base de données MOS ont été représentés sur la figure 4.8.

Les coefficients des différents hyperplans représentant les boîtes sont donnés dans le tableau 4.1. Nous rappelons qu'une boîte est définie par l'intersection de demi-espaces dont les bordures sont des hyperplans. Un hyperplan est défini, dans l'espace des critères, par son équation $a_1x_1 + a_2x_2 = 1$. Les coefficients a_1 et a_2 sont la première et la deuxième colonne des matrices ci-dessous, respectivement :

$$\underset{\text{boîte 0}}{\begin{pmatrix} 0,01 & 0 \\ 0 & 2 \\ 1 & 0,0077 \end{pmatrix}} \quad \underset{\text{boîte 1}}{\begin{pmatrix} 0,004 & 0 \\ 0 & 0,52 \\ 0,0033 & 0,33 \end{pmatrix}} \quad \underset{\text{boîte 2}}{\begin{pmatrix} 0,0025 & 0 \\ 0 & 0,25 \\ 0,0018 & 0,18 \end{pmatrix}}$$

TABLE 4.1 – Coefficients des hyperplans composant les différentes boîtes de la poupée russe

4.5.3 Méthodologie de comparaison

Dans ces expérimentations, nous voulons comparer les qualités des routages réalisés par la méthode de la poupée russe et par la méthode de la somme pondérée. Pour ce faire, le résultat d'une expérience de type MOS peut être considéré comme un arbitre qui décide quelle méthode est la meilleure. Cependant, les résultats obtenus à partir

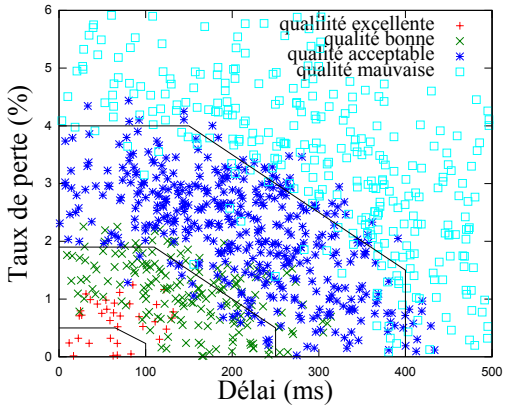


FIGURE 4.8 – Structure de la poupée russe pour l'application VoIP

d'un MOS sont aléatoires : pour un même vecteur de critères mesuré à destination, plusieurs utilisateurs peuvent donner différentes opinions subjectives. Par conséquent, la comparaison doit se faire sur la base d'une distribution de probabilité empirique, qui représente la qualité du routage. De ce fait, pour utiliser la notion de fonction de répartition empirique, les niveaux de qualité : excellent, bon, acceptable et mauvais sont représentés par une variable aléatoire entière prenant les valeurs : 3, 2, 1 et 0, respectivement. Deux distributions de probabilité $P\left(q_i \setminus \mathbf{c_i}\right)$ et $P\left(q_j \setminus \mathbf{c_j}\right)$ estimées à partir du résultat d'une expérience MOS peuvent être comparées au moyen du test statistique de Kolmogorov-Smirnov [Con80] pour déterminer avec un risque d'erreur donné si les échantillons de qualité $\{q_i\}$ and $\{q_j\}$ proviennent de la même distribution de probabilité. Dans ce cas, nous disons que les deux échantillons de qualité sont similaires. Dans le cas contraire, les distributions empiriques sont comparées en utilisant la dominance stochastique du premier ordre [HR69], définie comme suit : soient A et B deux distributions de probabilité, et F_A et F_B leurs fonctions de répartition respectives.

$$A \text{ “domine stochastiquement au premier ordre" } B \Leftrightarrow \begin{cases} \forall y, & F_B\left(y\right) \geqslant F_A\left(y\right) \\ \exists y, & F_B\left(y\right) > F_A\left(y\right) \end{cases}$$

Ainsi, lorsque A domine B, la fréquence des qualités inférieures à y associée à la distribution B est supérieure ou égale à la fréquence des qualités inférieures à y associée

à la distribution A, et ce pour toutes les valeurs de y.

Pour avoir une bonne estimation de $\{q_i\}$ il faudrait un grand nombre d'échantillons de qualité subjective q_i associés aux vecteurs de critères $\mathbf{c}_i$. Cependant, dans le cas général, la probabilité de tirer plusieurs fois le même vecteur de critères $\mathbf{c}_i$ durant une expérience MOS est égale à zéro, les coordonnées du vecteur de critères $\mathbf{c}_i$ étant définies dans des intervalles de $\mathbf{R}$. Pour remédier à cette difficulté, nous proposons d'estimer la densité de probabilité $P\{q_i \setminus \mathbf{c}_i\}$ au moyen de la méthode des k plus proches voisins (k-NN, *k-Nearest Neigbors*) [CH67] (Figure 4.9). La méthode k-NN exige la définition d'une distance utilisée dans l'espace des critères, ainsi que le choix de la valeur du paramètre k. Une distance euclidienne pondérée est utilisée dans ce contexte, elle est définie comme suit :

$$d(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^{n} \left(\frac{a_i - b_i}{u_i} \right)^2}$$

où u_i est la borne supérieure de la coordonnée i des vecteurs de critères contenus dans le MOS. La taille d'un échantillon $k = 128$ pour une base de données MOS contenant 10000 entrées a été déterminée au moyen de plusieurs expériences préliminaires, qui ont permis de trouver un bon compromis entre le biais et la variance des estimations.

Une simulation consiste à transmettre des paquets de données du nœud source au nœud destination pendant 100 secondes. Pour chaque paquet de données arrivant au niveau du nœud destination, un vecteur de critères $\mathbf{c}$ (délai et taux de perte moyens) est calculé.

Un histogramme de niveaux de qualité (Figure 4.10) est alors obtenu en appliquant la méthode d'estimation k-NN sur le MOS avec le vecteur $\mathbf{c}$ obtenu durant une simulation. Une comparaison de la qualité de routage nécessite une paire de simulations pour un vecteur de poids donné $\mathbf{w}$ associé à la méthode de la somme pondérée. L'une des simulations utilise la méthode de la poupée russe pour obtenir l'histogramme de qualité noté R. L'autre utilise la méthode de la somme pondérée dont l'histogramme de qualité associé est noté W. Les paramètres d_i sont choisis au hasard et sont identiques pour chaque paire de simulations.

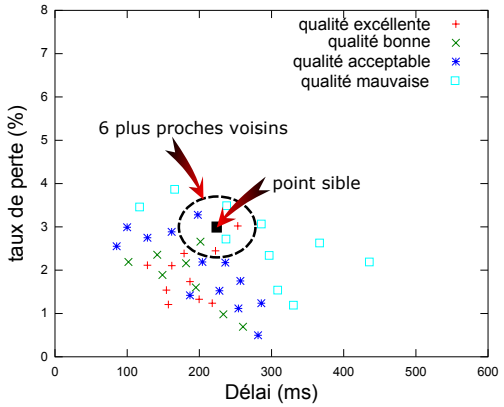


FIGURE 4.9 – Méthode des k plus proches voisins

Pour un vecteur de poids $\mathbf{w}$, N comparaisons de paires d'histogrammes sont effectuées. Les résultats sont les suivants :

- N_s : le nombre de comparaisons où les échantillons R et W sont similaires,
- N_r : le nombre de comparaisons où R domine W,
- N_w : le nombre de comparaisons où W domine R,
- N_n : le nombre de comparaisons où ni R ni W ne domine l'autre histogramme,

avec $N_s + N_r + N_w + N_n = N$. Nous définissons les fréquences relatives $f_s = N_s/N$, $f_r = N_r/N$, $f_w = N_w/N$ et $f_n = N_n/N$.

4.5.4 Résultats et discussion

Dans cette section, nous présentons trois expérimentations. La première a pour but de comparer les résultats obtenus par la méthode de la poupée russe et par celle de la somme pondérée sur un problème "jouet", en utilisant le MOS construit pour les problèmes de routage. Les deux autres expérimentations donnent des résultats de comparaison pour le problème de routage décrit précédemment.

Problème jouet

Dans cette expérimentation, le système de coordonnées de l'espace des critères est ramené entre 0 et 6 sur chaque axe des critères en divisant les délais par $100ms$ et en multipliant le taux de perte fois 100 pour tous les vecteurs de critère du MOS. Cette

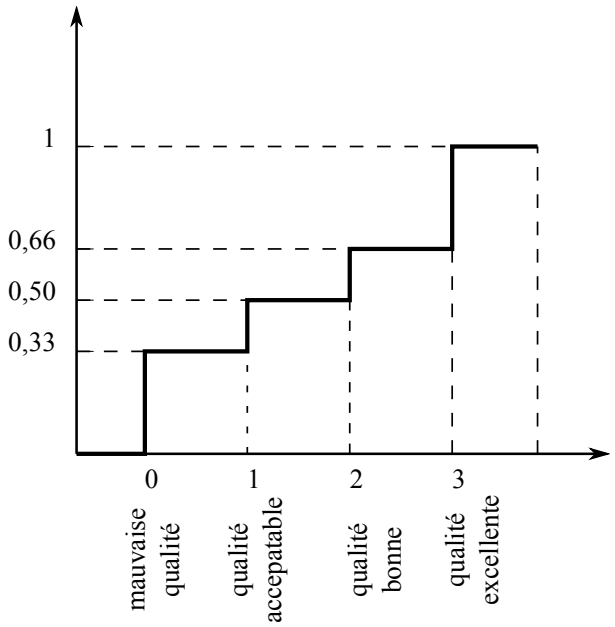


FIGURE 4.10 – Exemple d'une fonction de répartition d'un histogramme de qualité

mise à l'échelle est effectuée pour ramener les deux critères au même ordre de grandeur et faciliter l'analyse. Les méthodes de la poupée russe et de la somme pondérée sont testées au moyen d'un ensemble de solutions non-dominées situées sur un quart de cercle dont le rayon est égale à 1 et les coordonnées du centre sont des nombres réels choisis aléatoirement dans l'intervalle $[0, 6]$ (Figure 4.11). Les fréquences relatives f_s, f_r, f_w et f_n sont déterminées via une comparaison de 600 histogrammes de qualité générés pour chaque poids w_1 associé au critère délai et $w_2 = 1 - w_1$ associé au critère taux de perte. Les valeurs de f_s, f_r, f_w et f_n vs. w_1 sont présentées dans la figure 4.12. Cette dernière montre que les solutions trouvées par la méthode de la poupée russe dominent ou sont similaires à celles trouvées par la méthode de la somme pondérée pour presque toutes les comparaisons. Rarement, les histogrammes de qualité obtenus par la méthode de la somme pondérée dominent ceux fournis par la méthode de la poupée russe : au plus 3 cas sur 600 pour les valeurs de poids w_1 appartenant à l'intervalle $[0, 6; 0, 84]$. Nous avons vérifié que ces cas sont sporadiques. Ils sont probablement dus à des erreurs de première espèce commises par les tests de Kolmogorov-Smirnov. La figure montre également que les histogrammes de qualité pour cette expérimentation sont toujours comparables : $f_n = 0$ pour toutes les valeurs de poids.

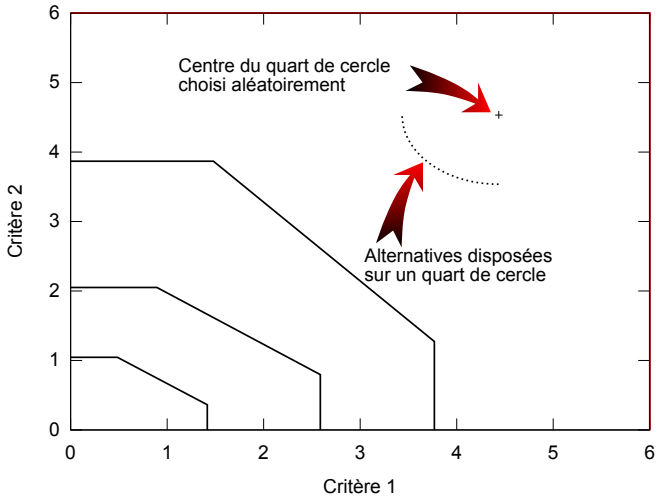


FIGURE 4.11 – Illustration de l'ensemble de solutions non-dominées utilisées dans le problème jouet

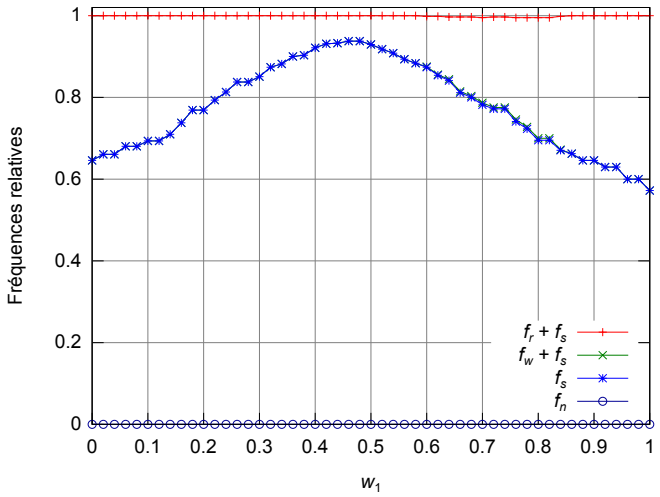


FIGURE 4.12 – Fréquences de dominance relatives dans le cadre du problème jouet : méthode de la poupée russe vs. méthode de la somme pondérée

Problème de routage multicritère avec le modèle *shadowing*

Dans cette section, nous présentons les résultats de simulation relatifs au scénario dans lequel les taux de perte l_i sont engendrés par le modèle *shadowing* (voir la section 4.5.1). La figure 4.14 montre les fréquences relatives $f_s + f_r$, $f_s + f_w$, f_s et f_n vs. w_1 obtenues à partir de 100 paires de simulations au niveau de la table de routage de la source. Nous rappelons que la table de routage de la source s contient les valeurs $\mathbf{Q}(a, d, s, n)$ qui représentent les espérances des vecteurs de critères à la destination

d associés à chaque voisin direct n et application a. Dans ce contexte, le vecteur de critères considéré pour le calcul des histogrammes de qualité R et W est celui donné par l'équation 4.3.

La figure 4.14 montre que les solutions trouvées par la méthode de la poupée russe dominent ou sont similaires à celles trouvées par la méthode de la somme pondérée pour toutes les valeurs de poids. Pour une seule paire de simulation parmi 100, la méthode de la somme pondérée domine la méthode de la poupée russe quand le poids w_1 est égal à $0,75$. Pour $w_1 = 0,5$ toutes les paires de simulations donnent des résultats similaires. Cela peut s'expliquer par le fait que la plupart des vecteurs optimaux obtenus avec la méthode de la poupée russe sont ceux qui sont les plus proches des hyperplans obliques, comme le montre la Figure 4.13. Dans ce cas, la méthode de la poupée russe se comporte comme une méthode de somme pondérée avec des poids égaux approximativement à $(0,5; 0,5)$ (voir chapitre 3). Pour les valeurs de poids différentes de $(0,5; 0,5)$, la fréquence des solutions similaires pour les deux méthodes décroît considérablement, dès que la valeur du poids du critère délai (w_1) est supérieure à $0,6$ ou inférieure à $0,45$.

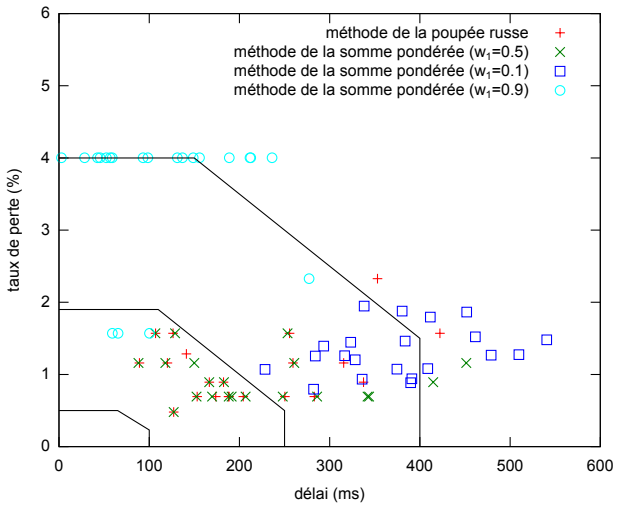


FIGURE 4.13 – Solutions trouvées par les deux méthodes : poupée russe et somme pondérée avec différents poids

Il est intéressant d'évaluer la qualité de routage à la destination, au lieu d'une prévision de qualité donnée par la table de routage du nœud source. La figure 4.15

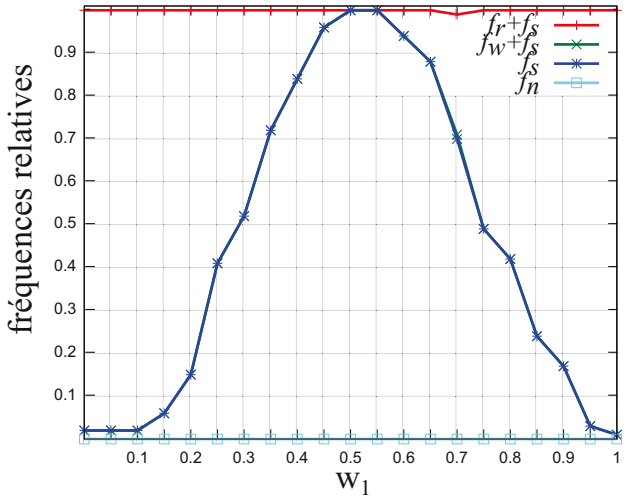


FIGURE 4.14 – Méthode de la poupée russe vs. méthode de la somme pondérée (au niveau de la source)

donne les fréquences relatives calculées à partir des vecteurs de critères évalués à la destination. Les figures 4.14 et 4.15 diffèrent parce que les valeurs des vecteurs de critères au niveau de la destination sont soumises à des variations aléatoires incessantes du délai et du taux de perte, qui sont imprévisibles. Malgré les variations stochastiques des critères, la méthode de la poupée russe domine nettement plus fréquemment la méthode de la somme pondérée pour les valeurs de w_1 en dehors de l'intervalle $[0,4; 0.6]$.

Problème de routage multicritère avec le modèle quadratique

Dans cette section, nous présentons les résultats de simulation relatifs au scénario dans lequel les taux de perte l_i sont générés avec une fonction quadratique (voir la section 4.5.1). La figure 4.16 renforce les résultats obtenus dans l'expérience précédente, car la grande majorité des solutions trouvées par la méthode de la poupée russe dominent ou sont similaires à celles trouvées par la méthode de la somme pondérée, et ce quelle que soit la configuration des poids. D'autre part, elle montre que la méthode de la poupée russe s'adapte aux variations de l'environnement, car elle trouve à chaque fois de meilleures solutions, sans avoir recours à une éventuelle reconfiguration de sa structure ou de ses paramètres. En effet, nous remarquons, à travers cette expérience, que les meilleurs routages trouvés par la méthode de la somme pon-

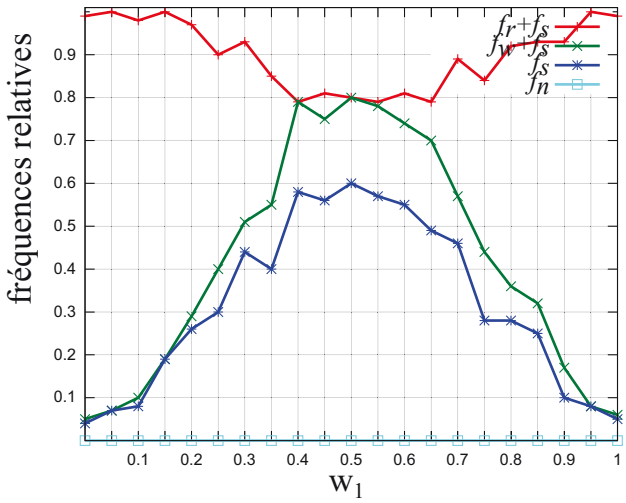


FIGURE 4.15 – Méthode de la poupée russe vs. méthode de la somme pondérée (au niveau de la destination)

dérée correspondent au cas où w_1 est dans l'intervalle $[0,40; 0,45]$ contrairement à la configuration optimale relative à l'expérience précédente, où w_1 est dans l'intervalle $[0,5; 0,55]$. Ainsi, si nous prenons une valeur de poids $w_1 = 0,52$ qui est le poids moyen optimal dans l'expérience relative au modèle *shadowing*, nous constatons que ce poids est notablement sous-optimal pour la somme pondérée dans l'expérience relative au modèle quadratique. En effet, la méthode de la poupée russe domine dans 30% des cas la somme pondérée avec $w_1 = 0,52$ dans l'expérience relative au modèle quadratique, tandis que ce taux de domination est de l'ordre de 10% pour le poids optimal. Cela montre que la méthode de la somme pondérée ne s'adapte pas à différentes conditions de fonctionnement du réseau. Les rares dominations de la méthode de la somme pondérée constatées figure 4.16 dans l'intervalle $[0,45; 0,55]$ pourraient être engendrées par des erreurs d'approximation de la base de donnée MOS pour le système de boîtes. La réponse précise à cette question nécessite des investigations supplémentaires.

4.6 Conclusion

Les réseaux ad hoc sans fil sont des réseaux complexes, qui sont capables de s'auto-organiser d'une façon autonome, afin de concevoir une topologie efficace pour la communication. La topologie est exploitée par un protocole de routage dédié qui établit

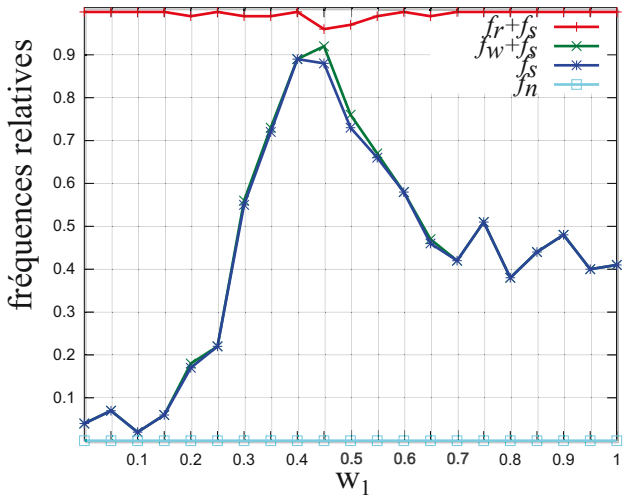


FIGURE 4.16 – Méthode de la poupée russe vs. méthode de la somme pondérée

des routes entre les différents nœuds, afin de faciliter la communication et de maximiser la qualité de service des applications qui transitent dans le réseau. Cependant, le protocole de routage doit alors tenir compte de la complexité de l'environnement, notamment l'aspect multicritère des exigences en QoS des différentes applications.

Dans ce chapitre, nous avons présenté un protocole de routage fondé sur l'apprentissage par renforcement utilisé conjointement avec une méthode de décision multicritère. Ce protocole facilite l'exploration de la topologie et la découverte de multitudes de routes multicritères entre la source et la destination.

Dans ce contexte, nous avons comparé la méthode de la poupée russe avec la méthode de la somme pondérée qui est très utilisée dans la littérature. Nous avons vu, à travers trois expérimentations, que la méthode de la poupée russe domine ou est similaire à la somme pondérée, dans presque 100% des simulations et ce, quelle que soit la configuration des poids utilisés par la méthode de la somme pondérée. De plus, l'utilisation de la somme pondérée, pour être performante, exige une recherche des poids optimaux, selon les variations de l'environnement. En revanche, les décisions prises avec la méthode de la poupée russe s'avèrent meilleures, quel que soit le contexte de fonctionnement. Ainsi, les résultats des comparaisons ont montré que la méthode de la poupée russe répond mieux que la méthode de la somme pondérée à l'aspect dynamique du routage dans un contexte autonome.

Chapitre 5

Conclusion

L'avènement des réseaux sans fil et l'explosion du nombre d'utilisateurs et d'applications rendent de plus en plus complexe l'environnement dans lequel opèrent les réseaux actuels. De ce fait, les chercheurs sont souvent appelés à développer de nouvelles solutions et protocoles qui s'adaptent à l'état de l'environnement de ces réseaux, qui est souvent dynamique. Au début, les chercheurs ont pensé à modifier les protocoles existants pour les rendre de plus en plus réactifs aux changements rapides de leur environnement. Cependant, cette approche a donné naissance à des architectures et protocoles complexes, qui sont difficiles à gérer. Ensuite, les chercheurs ont eu l'idée de reproduire le mécanisme de fonctionnement des systèmes autonomes qui s'adaptent à leur environnement via une boucle de rétroaction. Cette approche a donné naissance aux réseaux dits autonomes ou cognitifs. Les réseaux autonomes utilisent une boucle de rétroaction pour observer, planifier, décider puis agir, en fonction des événements qu'ils reçoivent de leur environnement [Mah07]. De plus, les réseaux autonomes doivent apprendre du résultat de leurs décisions pour corriger leur stratégie future. Ainsi, les réseaux autonomes apprennent en continu leur environnement, afin de converger de proche en proche vers un fonctionnement optimal.

Le domaine des réseaux autonomes a été traité particulièrement en détail dans le volet apprentissage et raisonnement ; cependant, il y a peu de travaux qui se sont intéressés à l'aspect multicritère du mécanisme de décision, qui demeure une problématique importante dans ce type de réseaux. Dans ce contexte, la plupart des méthodes

de décision multicritère classiques introduites dans la littérature sont inadaptées aux réseaux autonomes, car elles présentent l'un ou les deux inconvénients suivants :

- Elles nécessitent la présence d'un décideur qui doit formaliser ses préférences selon une méthodologie prédéfinie. Cependant, l'aspect autonome d'un système suppose l'absence d'interactions avec un opérateur humain.
- Elles utilisent des paramètres difficiles à déterminer. Généralement, la plupart des méthodes de décision multicritère utilisent des poids de quantification de l'importance d'un critère dans le processus de décision multicritère. Cependant, il n'existe aucune méthode qui garantit une correspondance optimale entre l'importance d'un critère dans le processus de décision et la valeur d'un poids de quantification.

Dans cette thèse, nous nous sommes particulièrement focalisés sur le développement d'une nouvelle méthode de décision multicritère, qui répond aux problématiques citées ci-dessus.

5.1 Résumé des contributions

Les principales contributions de cette thèse se résument comme suit :

- Nous avons proposé une nouvelle méthode de décision multicritère adaptée aux systèmes autonomes en général et aux réseaux autonomes en particulier. La nouvelle méthode, dite "méthode de la poupée russe", permet trois types d'adaptations :

  1. Elle peut adapter l'importance des critères selon leurs valeurs grâce aux multiples boîtes de qualité englobantes.
  2. Elle offre la possibilité de prendre en compte les différentes relations qui existent entre critères, en l'occurrence la compensation (boîtes de type simplexe), la non-compensation (boîtes de type hyperrectangulaire) et une combinaison de ces dernières (intersection d'un simplexe et d'un hyperrectangle).
  3. Elle offre la possibilité d'apprendre ses paramètres à partir de données dont

l'acquisition peut être automatisée, ce qui lui permet d'adapter sa forme si cela est utile à la maximisation des performances du système.

- Nous avons élaboré un protocole de routage fondé sur l'apprentissage par renforcement utilisé conjointement avec une méthode de décision multicritère. Ce protocole facilite l'exploration de la topologie et la découverte de multiples routes multicritères entre source et destination.
- Nous avons comparé les performances de la méthode de la poupée russe face à celles de la méthode de la somme pondérée, qui est la méthode la plus utilisée dans la littérature pour réaliser un routage multicritère dans un réseau ad hoc sans fil. Nous avons constaté que la méthode de la poupée russe surpasse nettement la méthode de la somme pondérée dans la quasi-totalité des configurations des poids. En effet, la méthode de la somme pondérée fournit au mieux les mêmes performances que la méthode de la poupée russe.

5.2 Perspectives et travaux futurs

Étant donné que ce travail s'inscrit dans un domaine encore peu exploré qui traite de la décision multicritère dans les réseaux autonomes, il reste du chemin à parcourir afin d'affiner l'approche proposée. Plus précisément, nous proposons de poursuivre le travail réalisé à travers les directions de recherche suivantes :

- Comparer la méthode de la poupée russe à d'autres méthodes de décision multicritère, notamment les méthodes à préférences *a priori*,
- Étudier en détail la relation entre critères/métriques présentes dans le domaine des réseaux sans fil, afin de trouver une correspondance optimale entre la structure de la poupée russe et la relation entre critères,
- Étudier le comportement de la méthode de la poupée russe sur des problèmes de décision multicritère sous contraintes. En effet, le formalisme devrait être bien adapté à la prise en compte de contraintes,
- Étudier en détail le protocole de routage multicritère fondé sur l'apprentissage par renforcement, afin de traiter toutes les problématiques relatives aux réseaux

ad hoc sans fil, notamment la mobilité,

– Tester la méthode de la poupée russe combinée avec le protocole de routage basé sur l'apprentissage par renforcement dans un cas réel de réseau ad hoc sans fil,

Références bibliographiques

[AEES08] N.A. Ali, E. Ekram, A. Eljasmy, and K. Shuaib. Measured delay distribution in a wireless mesh network test-bed. In *IEEE/ACS International Conference on Computer Systems and Applications*, pages 236–240, Doha, Qatar, April 2008.

[AP96] T.W. Athan and Y.P. Panos. A note on weighted criteria methods for compromise solutions in multi-objective optimization. *Engineering Optimization*, 27(2) :155–176, 1996.

[BDM08] J. Branke, K. Deb, and K. Miettinen. *Multiobjective optimization : Interactive and evolutionary approaches*, volume 5252. Springer-Verlag New York Inc, 2008.

[BEF+00] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, et al. Advances in network simulation. *Computer*, 33(5) :59–67, 2000.

[BMV78] J.P. Brans, B. Mareschal, and Ph. Vincke. A new family of outranking methods in multi-criteria analysis. *Cahiers du Centre d'études de Recherche Opérationnelle*, pages 3–24, 1978.

[CC61] A. Charnes and W.W. Cooper. *Management models and industrial applications of linear programming.* John Wiley & Sons, 1961.

[CH67] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1) :21–27, 1967.

[Con80] W.J. Conover. Practical nonparametric statistics. *J. Wiley & Sons*, 1980.

[CS02] Y. Collette and P. Siarry. *Optimisation multiobjectif*. EYROLLES, 2002.

[CSMS04] P. Calyam, M. Sridharan, W. Mandrawa, and P. Schopis. Performance measurement and analysis of h.323 traffic. In Chadi Barakat and Ian Pratt, editors, *Passive and Active Network Measurement*, volume 3015 of *Lecture Notes in Computer Science*, pages 137–146. Springer Berlin / Heidelberg, 2004.

[DAPM00] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization : NSGA-II. In *Parallel Problem Solving from Nature PPSN VI*, pages 849–858. Springer, 2000.

[Deb01] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, 2001.

[DPST03] J. Dréo, A. Pétrowski, P. Siarry, and E. D. Taillard. *Métaheuristiques pour l'optimisation difficile*. Eyrolles, September 2003.

[DS06] K. Deb and J. Sundar. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 635–642, New York, USA, July 2006. ACM.

[ED06] K. Egoh and S. De. A Multi-Criteria Receiver-Side Relay Election Approach in Wireless Ad Hoc Networks. In *Military Communications Conference*, pages 1–7, Washington D.C., USA, oct. 2006.

[Ehr05] M. Ehrgott. *Multicriteria optimization*, volume 491. Springer Verlag, 2005.

[FF+93] C.M. Fonseca, P.J. Fleming, et al. Genetic algorithms for multiobjective optimization : Formulation, discussion and generalization. In *Proceedings of the fifth international conference on genetic algorithms*, volume 423, pages 416–423, Urbana-Champaign, USA, July 1993.

[Fis70] Fishburn. *Utility theory for Decision Making*. John Wiley & Sons, 1970.

[GBR06] X. Gandibleux, F. Beugnies, and S. Randriamasy. Martins' algorithm revisited for multi-objective shortest path problems with a MaxMin cost function. *4OR : A Quarterly Journal of Operations Research*, 4(1) :47–59, 2006.

[GDRMB09] F. Guerriero, F. De Rango, S. Marano, and E. Bruno. A biobjective optimization model for routing in mobile ad hoc networks. *Applied Mathematical Modelling*, 33(3) :1493–1512, 2009.

[Gol89] D.E. Goldberg. *Genetic Algorithms for Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, 1989.

[GR87] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 41–49, Hillsdale, NJ, USA, 1987.

[HNG94] J. Horn, N. Nafpliotis, and D.E. Goldberg. A niched Pareto genetic algorithm for multiobjective optimization. In *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pages 82–87, Urbana, USA, Jun 1994.

[Hor97] J. Horn. Multicriteria decision making and evolutionary computation. In *Handbook of Evolutionary Computation*. Institute of Physics Publishing, London, 1997.

[HR69] J. Hadar and W.R. Russell. Rules for ordering uncertain prospects. *The American Economic Review*, 59(1) :25–34, 1969.

[Ish06] Nojima Y. Narukawa K. Doi T. Ishibuchi, H. Incorporation of decision maker's preference into evolutionary multiobjective optimization algorithms. In *GECCO'06*, pages 741–742, New York, USA, July 2006.

[Kla09] K. Klamroth. Discrete multiobjective optimization. In Matthias Ehrgott, Carlos Fonseca, Xavier Gandibleux, Jin-Kao Hao, and Marc Sevaux, editors, *Evolutionary Multi-Criterion Optimization*, volume 5467 of *Lecture Notes in Computer Science*, pages 4–4. Springer Berlin / Heidelberg, 2009.

[KR76] Ralph L. Keeney and Howard Raiffa. *Decisions with Multiple Objectives : Preferences and Value Tradeoffs*. Wiley series in probability and mathematical statistics. John Wiley & Sons, Inc, New York, 1976.

[MA10] R. Marler and Jasbir Arora. The weighted sum method for multi-objective optimization : new insights. *Structural and Multidisciplinary Optimization*, 41 :853–862, 2010.

[Mah07] Q.H. Mahmoud. *Cognitive networks*. Wiley Online Library, 2007.

[Mar84] E.Q.V. Martins. On a multicriteria shortest path problem. *European Journal of Operational Research*, 16(2) :236–245, 1984.

[Mie99] K. Miettinen. *Nonlinear multiobjective optimization*, volume 12. Springer, 1999.

[Mit00] J. Mitola. *Cognitive Radio : An Integrated Agent Architecture for Software Defined Radio*. PhD thesis, Royal Inst. Technology, Sweden, 2000.

[MRA+09] A.B. MacKenzie, J.H. Reed, P. Athanas, C.W. Bostian, R.M. Buehrer, L.A. DaSilva, S.W. Ellingson, Y.T. Hou, M. Hsiao, J.M. Park, et al. Cognitive radio and networking research at virginia tech. *Proceedings of the IEEE*, 97(4) :660–688, 2009.

[MTT+06] B. Malakooti, I. Thomas, S.K. Tanguturi, S. Gajurel, H. Kim, and K. Bhasin. Multiple criteria network routing with simulation results. In *Proc. 15th Industrial Engineering Research Conference (IERC)*, Orlando, USA, May 2006.

[Per02] C. Perkins. IP mobility support for IPv4 (RFC 3344). *Internet Engineering Task Force (IETF)*, 2002.

[RB73] B. Roy and P. Bertier. La méthode ELECTRE II Une application au média planning. *OR'72, ed. M. Ross, Amsterdam : North-Holland Publishing Company*, pages 291–302, 1973.

[RB93] B. Roy and D. Bouyssou. *Aide Multicritère à la Décision : Méthodes et Cas*. Economica, 1993.

[RD04] A. Roy and S.K. Das. QM2RP : A QoS-based mobile multicast routing protocol using multi-objective genetic algorithm. *Wireless Networks*, 10 :271–286, 2004.

[REM08] C.S. Randriamasy and Y. El Mghazli. Device for determining switching paths in a label switched communication network in the presence of selection attributes. Patent, 10 2008. US 7443832.

[Roy68] B. Roy. Classement et choix en présence de points de vue multiples. *Revue Française d'Informatique et de Recherche Opérationnelle*, (8) :57–75, 1968.

[Roy78] B. Roy. ELECTRE III : Un algorithme de classements fondé sur une représentation floue des préférences en présence de critères multiples. *Cahiers du Centre d'études de Recherche Opérationnelle*, pages 3–24, 1978.

[RR89] J.R. Rao and N. Roy. Fuzzy set theoretic approach of assigning weights to objectives in multicriteria decision making. *Int J Syst Sci 20*, pages 1381–1386, 1989.

[RV89] B.P. Roy and P. Vincke. *L'aide multicritère à la décision.* Editions Ellipses, 1989.

[Saa80] T.L. Saaty. *The Analytical Hierarchy Process.* McGraw-Hill, 1980.

[Saa08] T.L. Saaty. Decision making with the analytic hierarchy process. *International Journal of Services Sciences*, 1(1) :83–98, 2008.

[SAZ07] S.R. Saunders and A. Aragón-Zavala. *Antennas and propagation for wireless communication systems.* Wiley, 2007.

[Sch85] J. David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 93–100, Hillsdale, NJ, USA, 1985.

[Sch03] H. Schulzrinne. RTP : A Transport Protocol for Real-Time Applications (RFC 3550). *Internet Engineering Task Force (IETF)*, 2003.

[SD94] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3) :221–248, 1994.

[SH98] T.L. Saaty and G. Hu. Ranking by eigenvector versus other methods in the analytic hierarchy process. *Appl Math Lett 11*, pages 121–125, 1998.

[SK06] Miettinen K. Vuori J. Setmaa-Krkkinen, A. Best compromise solution for a new multiobjective scheduling problem. *Comput. Oper. Res.33*, pages 2353–2368, 2006.

[Sut98] Barto A. Sutton, R. *Reinforcement Learning : An Introduction.* MIT Press, Cambridge, Massachusetts, 1998.

[TFDM06] R.W. Thomas, D.H. Friend, L.A. DaSilva, and A.B. MacKenzie. Cognitive networks : adaptation and learning to achieve end-to-end performance objectives. *Communications Magazine, IEEE*, 44(12) :51–57, 2006.

[Uni96] International Telecommunication Union. ITU-T Rec. P.800 : methods for Subjective Determination of Transmission Quality. *Telecommunication standardization sector*, 1996.

[Vin92] Ph. Vincke. *Multicriteria Decision-Aid.* John Wiley & Sons, 1992.

[WC96] Z. Wang and J. Crowcroft. Quality-of-service routing for supporting multimedia applications. *Selected Areas in Communications, IEEE Journal on*, 14(7) :1228–1234, 1996.

[Wie80] A. P. Wierzbicki. The use of reference objectives in multiobjective optimization. *In G. Fandel and T. Gal, editors, Multiple Criteria Decision Making Theory and Applications*, pages 468–486, 1980.

[YH95] K. Yoon and C.L. Hwang. *Multiple attribute decision making : an introduction.* Sage Publications Thousand Oaks, CA, 1995.

[Zad63] L. Zadeh. Optimality and non-scalar-valued performance criteria. *Automatic Control, IEEE Transactions on*, 8(1) :59 – 60, jan 1963.

[ZLT02] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2 : Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In K.C. Giannakoglou et al., editors, *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE), 2002.

[ZT98] E. Zitzler and L. Thiele. An evolutionary algorithm for multiobjective optimization : The strength Pareto approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zürich, Switzerland, 1998.