

Un système multi-agents pour une place de marché de facture

Cédric Buron

▶ **To cite this version:**

Cédric Buron. Un système multi-agents pour une place de marché de facture. Intelligence artificielle [cs.AI]. Sorbonne Université, 2018. Français. NNT : 2018SORUS314 . tel-01900075v2

HAL Id: tel-01900075
https://theses.hal.science/tel-01900075v2

Submitted on 11 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.





THÈSE DE DOCTORAT DE
SORBONNE UNIVERSITÉ

Spécialité

Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Cédric Buron

Pour obtenir le grade de

DOCTEUR de SORBONNE UNIVERSITÉ

Sujet de la thèse :

Un Système Multi-Agents pour une place de marché de factures

soutenue le 17 mai 2018

devant le jury composé de :

Mme Zahia	Guessoum	Directrice de thèse
M. Sylvain	Ductor	Co-encadrant
M. Olivier	Roussel	Co-encadrant
Mme Maroua	Bouzid-Mouaddib	Rapporteure
M. René	Mandiau	Rapporteur
Mme Amal	El-Fallah Seghrouchni	Examinatrice
M. Vincent	Chevrier	Examinateur

Résumé

Dans les entreprises, l'écart entre la date où les entreprises engagent leurs dépenses et la date où elles sont payées entraîne un besoin en fonds de roulement à l'origine de problèmes dont la sévérité va de la perte de contrat au dépôt de bilan. Dans le contexte actuel, le recours à l'emprunt pour éviter ces situations est hélas complexe pour les PME, les banques refusant de prêter lorsqu'elles estiment que le risque est trop important. Une possibilité est alors de recourir au factoring, et de faire financer ses créances.

Dans cette thèse, nous proposons une place de marché de factoring intelligente, basée sur des travaux de négociation automatique et de mechanism design : nous proposons d'analyser théoriquement et expérimentalement les problématiques liées aux agents curieux cherchant à inférer les informations privées des autres agents, entraînant des surcoûts de leur point vue comme de celui de la plate-forme. Nous proposons également de concevoir un protocole de négociation qui s'oppose à de tels comportements, donnant aux agents une incitation à ne négocier que lorsqu'ils souhaitent vraiment obtenir un bien.

Nous proposons ensuite un agent de négociation automatique. Pour la conception de la stratégie de notre agent, nous proposons d'exploiter les méthodes de Monte Carlo Tree Search, qui ont récemment fait leur preuve dans l'IA pour les jeux. Notre agent est capable d'apprendre de son partenaire bien que ce ne soit pas son objectif premier, afin de trouver un accord qui lui soit favorable. Il s'appuie pour cela sur des technique de modélisation d'adversaire impliquant des méthodes d'apprentissage tels que l'apprentissage bayésien et la régression de processus gaussiens.

Abstract

For most companies, the time gap between the moment when they begin to spend money and the moment when they receive payment from their customer generates a working capital requirement. The latter may create issues such as contract lost or bankruptcy. In the current context, getting credits from banks may be either costly or even impossible for Small and Medium Enterprises. Indeed, many banks refuse to lend money for small companies or when they estimate that there is a risk. It is then possible to rely on other solutions as factoring and get immediate credit from the invoices.

In this thesis, we propose a factoring marketplace based on related to curious agents that try to infer private information from their partners. This kind of behavior is harmful both for the agents and for the platform more generally. We thus propose to design a negotiation protocol resistant to such behavior by endowing the agents with an incentive to negotiate only when they really try to get a good.

We then propose an automated negotiation agent which relies on Monte Carlo Tree Search techniques. Those techniques have proved to be quite efficient in AI for games. Our agent is able to learn information from its partner, though it is not its main objective in order to get to a beneficial agreement. For this purpose, it relies on opponent modeling techniques and machine learning such as Bayesian Learning and Gaussian Process Regression.

Remerciements

Bien que cette thèse soit le fruit de mon travail, elle doit en fait beaucoup à toutes les personnes qui m'ont aidé, soutenu et critiqué mon travail au cours des plus de trois ans qu'ont constitués mon doctorat. Je tiens tout d'abord à remercier du fond du cœur Zahia. La patience, la disponibilité et la confiance qu'elle m'a accordées tout au long de cette période m'ont permis de mener à bien ce travail dans les meilleures conditions. Elle a non seulement été une directrice de thèse et une encadrante hors du commun, mais aussi une oreille à l'écoute des difficultés et des doutes que j'ai expérimentés aussi bien que des idées que j'ai proposées. Il est rare de rencontrer des personnes faisant passer l'intérêt d'une autre personne avant la leur. Zahia est de celles-là.

Je tiens aussi à remercier mes deux co-encadrants, Sylvain et Olivier, sans l'exigence desquels mon travail eût été d'une bien moindre qualité. Je dois aux nombreuses discussions que nous avons eues un certain nombre d'idées de mon travail. Ils ont aussi relu tout ou partie de cette thèse, corrigeant mes erreurs et commentant de manière toujours pertinentes ce manuscrit autant que les publications que j'ai réalisées lors de mon doctorat. Cette thèse leur doit beaucoup. Au-delà du cadre professionnel et scientifique, les discussions que nous avons eues ont nourri ma curiosité sur des sujets parfois bien loin du sujet de mon doctorat. Si ces années ont marqué un tel changement dans ma perception du monde, je le dois en partie à ces discussions.

Un immense merci à Maroua Bouzid-Mouaddib et René Mandiau, qui ont accepté de relire mon travail, et m'ont fait un retour juste et pertinent. On ne se rend pas compte, avant de l'avoir devant les yeux, à quel point une revue peut être à la fois précieuse et gratifiante. Je tiens à remercier René Mandiau également pour les très nombreux commentaires qu'il a faits en vue de la corrections de toutes les petites erreurs qui m'ont échappées, et qui amélioreront la qualité de ce travail. Merci aussi à Vincent Chevrier et à Amal El Fallah Seghrouchni pour avoir accepté de participer au jury de cette thèse.

Une thèse CIFRE a pour particularité de se dérouler dans deux cadres à la fois. Dans mon cas, il s'agit du Laboratoire d'Informatique de Paris 6 et de Kyriba. Les personnes que j'ai rencontrées et avec qui j'ai collaboré dans ces deux milieux ont été d'un grand secours durant ce doctorat. Au sein du LIP6, je tiens en particulier à remercier Cédric Herpson pour m'avoir confié ses étudiants de L2, et plus généralement pour sa constante bienveillance. Merci aussi à Nicolas Maudet, Vincent Corruble, Jean-Pierre Briot pour être venus assister à la présentation de mes travaux, et pour vos remarques constructives. Merci aussi à Jean-Daniel Kant pour m'avoir invité à l'atelier MAGECO qui m'a ouvert de nombreuses perspectives et m'a fait voir un autre pan des SMA. Merci à Christophe Bouder, Vincent Cuzin, Jean-Pierre Arranz et Belaïd Naït Sidnas pour leur soutien technique et administratif lorsque j'en ai eu besoin, et qui m'a permis de me focaliser sur la recherche.

Enfin, merci à tous mes amis stagiaires, doctorants et post doc avec lesquels j'ai eu des discussions passionnantes sur l'intelligence artificielle ou sur tant d'autres sujets, et en particulier Mathieu, Dounia, Siao-Leu, Medhi, Ludovic, Anastasia, Kenza, Loïs, et un grand grand merci à Karim, qui m'a sauvé à de nombreuses reprises sur le plan administratif, quand les dates se faisaient pressantes.

Merci aussi à ceux qui m'ont aidé et soutenu à Kyriba. À l'équipe Recherche et Développement,

qui m'a accueilli à bras ouverts, et avec lesquels j'ai partagé trois années. Je ne peux pas tous les remercier individuellement. J'espère simplement rendre justice à certains d'entre eux : Laure, Sonia, Arjun, Nicolas, Henri, François, Louay, Attilla, Philippe, Christophe, Youssef, Maxime, Kieran, Alexey, Frédéric, Ilya, Tianlin, Brice... Merci à Seng, Guillaume et Emmanuel qui m'ont sauvé la veille d'une démonstration, en me débloquant sur le logiciel. Merci aussi à ceux qui m'ont entouré, et dont les remarques constructives ont fait évoluer mon travail : les Fraudettes, Emmanuel, Vincent, Xavier A., Xavier T. et Tanguy, Jamel et Armel.

Un immense merci à la team Sakura et à la team Île O Crêpes, qui se reconnaîtront, ainsi qu'à ceux avec qui j'ai déjeuné tant et tant de fois au troisième, et qui avec lesquels j'ai eu des discussions allant du fou rire au débat passionnant : Véro, Émilie, Sandrine, Charlotte, M-C, Mamou, Pirom, Céline, Thomas, Stéphane, David... Un immense merci aussi à mes camarades de la DUPr.

Ce doctorat a été réalisé au sein d'un consortium FUI, et je tiens à remercier ses membres qui m'ont appris beaucoup, et qui m'ont aidé au moment où il a fallu rendre compte de ce projet. En particulier, merci à Valentin Schnapper pour son aide sur les livrables, à Nabil pour ses explications sur la partie risque et assurance, à Gilles pour les discussions que nous avons eues sur l'apprentissage automatique, à Jean Laurent Viviani et à Vincent Hovelaque pour leur dynamisme et pour avoir organisé des doctoriales ainsi qu'aux membres du comité de suivi, en particulier Véronique Dalfarra pour sa bienveillance et sa rigueur.

Merci à mes collègues de Thales, qui m'ont accueilli et m'ont fait confiance alors même que je n'avais pas encore envoyé mon manuscrit : Florence et Christophe en premier lieu, tous mes collègues de LDO et des autres laboratoires de STI, à Alexandru Enache et Cédric Demeure.

Mon parcours doit beaucoup aux expériences de recherche que j'ai faites avant de commencer. Je tiens donc à remercier ceux qui m'ont aidé à développer ce goût : Élisabeth Brunet, Tristan Cazenave, Wu I-Chen, Abdallah Saffidine, et Marek Cornu.

Cette thèse doit aussi beaucoup à ses relecteurs, en particulier mon père, qui en a relu l'intégralité et a corrigé mille et une erreurs, et mon frère qui a relu la partie mathématique et a fait des remarques très pertinentes, qui m'ont beaucoup aidé.

Enfin, on a tendance à penser qu'un doctorat est une question d'intelligence avant tout. C'est faux. C'est d'abord une question de volonté, de moral. Je tiens donc à remercier ma famille et mes amis, qui m'ont énormément soutenu durant ces trois ans. Ma femme tout d'abord, qui a au quotidien partagé mes moments de doute et mes remises en question. Merci du fond du cœur. Mes parents et mes frères aussi, qui leur ont prêté une oreille attentive. Enfin, tous mes amis qui m'apportent leur soutien et à qui je dois tant. Merci à vous d'être là.

Table des matières

Introduction **1**
1 Présentation du problème . . . 1
2 Objet de cette thèse . . . 4
3 Organisation de cette thèse . . . 5

1 Places de marché multi-agents **7**
1.1 Introduction . . . 8
1.2 Étude des places de marché . . . 9
1.2.1 Clients des places de marché . . . 9
1.2.2 Protocoles . . . 16
1.2.3 Objet de la négociation . . . 17
1.2.4 Préférences . . . 17
1.2.5 Places de marché existantes . . . 18
1.3 Notre plate-forme multi-agents pour le commerce de créances . . . 22
1.3.1 OntoMASK . . . 23
1.3.2 Considérations techniques . . . 28
1.3.3 Discussion . . . 31
1.4 Conclusion . . . 32

2 Protocoles de négociation **35**
2.1 Introduction . . . 36
2.2 Protocoles de négociation . . . 37
2.2.1 Enchères . . . 37
2.2.2 *Bargaining* . . . 42
2.2.3 Réseaux contractuels . . . 44
2.2.4 Modèle producteurs-consommateurs . . . 46
2.2.5 Discussion . . . 52
2.3 *Mechanism design* . . . 53
2.3.1 Principes . . . 53
2.3.2 *Mechanism design* et *bargaining* . . . 58
2.3.3 Discussion . . . 64
2.4 Conclusion . . . 65

3 Agents négociateurs **67**
3.1 Introduction . . . 68
3.2 Modélisation de l'adversaire . . . 69
3.2.1 Stratégie d'offres . . . 70
3.2.2 Le prix de réserve . . . 74

3.2.3 La fonction d'utilité . . . 75
3.2.4 Stratégie d'acceptation . . . 77
3.2.5 Deadline . . . 78
3.2.6 Discussion . . . 79
3.3 Stratégies d'offres . . . 79
3.3.1 Stratégies fixes . . . 79
3.3.2 Stratégies adaptatives utilisant des stratégies fixes . . . 83
3.3.3 Intervalles . . . 85
3.3.4 Algorithmes génétiques . . . 85
3.3.5 En utilisant les propositions de l'adversaire . . . 87
3.3.6 Discussion . . . 87
3.4 Stratégie d'acceptation . . . 88
3.4.1 Stratégies simples . . . 88
3.4.2 Combinaison de stratégies simples . . . 89
3.4.3 Stratégie optimale . . . 89
3.4.4 Courbe de concession . . . 89
3.4.5 Discussion . . . 90
3.5 Conclusion . . . 90

4 Agents curieux **93**
4.1 Introduction . . . 94
4.2 Sensibilité à la curiosité . . . 94
4.2.1 Problématique . . . 95
4.2.2 Formalisation . . . 95
4.2.3 Taxonomie des agents . . . 98
4.2.4 Discussion . . . 101
4.3 Protocole résistant aux agents curieux . . . 101
4.3.1 *Bargaining* et agents curieux . . . 101
4.3.2 Cas mono-attribut : négociation sur le prix . . . 102
4.3.3 Cas multi-attributs . . . 107
4.3.4 Discussion . . . 111
4.4 Implémentation . . . 111
4.4.1 Implémentation centralisée . . . 111
4.4.2 Implémentation décentralisée . . . 112
4.4.3 Implémentation 1:n . . . 115
4.4.4 Analyse expérimentale . . . 116
4.4.5 Discussion . . . 121
4.5 Conclusion . . . 121

5 MoCaNA : Un agent de négociation basé sur MCTS **123**
5.1 Introduction . . . 124
5.2 Problématique . . . 125
5.2.1 Méthodes de Monte Carlo . . . 126
5.2.2 Problématiques spécifiques à la négociation . . . 130
5.2.3 Discussion . . . 131
5.3 Un *Monte Carlo-based Negotiation Agent* (MoCaNA) . . . 132
5.3.1 Modélisation d'adversaire . . . 133
5.3.2 Une stratégie d'offre basée sur MCTS . . . 144
5.3.3 Stratégie d'acceptation . . . 146
5.3.4 Discussion . . . 146

5.4 Analyse expérimentale . . . 147
5.4.1 Détails d'implémentation . . . 147
5.4.2 Protocole expérimental . . . 149
5.4.3 Résultats . . . 150
5.4.4 Discussion . . . 153
5.5 Conclusion . . . 154

Conclusion **155**
1 Introduction . . . 155
2 Contributions . . . 155
2.1 Agents curieux . . . 156
2.2 MoCaNa . . . 156
3 Perspectives . . . 157
3.1 Agents curieux . . . 157
3.2 MoCaNA . . . 158

Bibliographie **159**

Table des figures

1 Architecture du projet . . . 3
2 Organisation des différents éléments de la négociation . . . 5

1.1 Le modèle CBB . . . 10
1.2 Le modèle BCP, tel que décrit par [Chen *et al.*, 2008] . . . 12
1.3 Exemple d'*Analytic Hierarchic Tree* . . . 14
1.4 Le modèle BBT . . . 15
1.5 Protocoles de négociation multi-agents . . . 16
1.6 Hiérarchie des agents . . . 26
1.7 Hiérarchie des acteurs . . . 26
1.8 Hiérarchie des biens . . . 26
1.9 Ontologie de la plate-forme . . . 27
1.10 Architecture applicative d'une plate-forme Kyriba App . . . 29
1.11 Architecture physique et organisation des différentes plates-formes . . . 30
1.12 Architecture fonctionnelle et organisation des modules . . . 31
1.13 Architecture physique et modifications amenées par la plate-forme . . . 32

2.1 Deux variantes de *bargaining* multilatéral . . . 43
2.2 Diagramme de séquences du *bargaining* multitiers . . . 44
2.3 Diagramme de séquences du protocole de réseau contractuel . . . 45
2.4 Diagramme séquences de l'*Iterated Contract Net Protocol* . . . 46
2.5 Principe de révélation . . . 58
2.6 Espace réalisable et espace acceptable . . . 59
2.7 Espace convexe d'optimisation . . . 63

3.1 Modèle BOA . . . 69
3.2 Principe de la régression non linéaire . . . 71
3.3 Exemple de processus gaussien . . . 73
3.4 Modélisation de stratégie par réseau de neurones . . . 74
3.5 Représentation de la modélisation de stratégie par un apprentissage Bayésien . . . 77
3.6 Interpolation d'une fonction par n polynomes de Tchebychev . . . 78
3.7 Influence de β sur la stratégie polynomiale pour un acheteur . . . 81
3.8 Algorithme génétique de GANGSTER . . . 86

4.1 Exemple de négociation . . . 97
4.2 Stratégie et prix de réserve des différents types d'agents . . . 100
4.3 Frontière d'un ensemble . . . 108
4.4 Existence de $\mathbb{O}_{12}$ . . . 109
4.5 Les trois cas possibles de la démonstration du Théorème 4.6 . . . 110

4.6 Implémentation centralisée du protocole résistant aux agents curieux 112
4.7 Implémentation décentralisée de l'échange de messages 115
4.8 Illustration de l'implémentation 1:n 116
4.9 Deux exemples de négociations et prix de réserve pour le *bargaining* mono-attribut (prix) 117
4.10 Impact de la borne sur le bien-être des différents types d'agents 119
4.11 Bien-être des agents, dans différentes situations 120

5.1 Calcul d'intégrale par la méthode de Monte-Carlo 126
5.2 Algorithme MCTS 129
5.3 Interaction des modules de MoCaNA 132
5.4 Exemple de négociation multi-attribut 133
5.5 Densité de probabilité d'une loi normale bivariée 135
5.6 Résultat d'une régression de processus gaussien 137
5.7 Représentation des *kernels* 138
5.8 Fonction logit 140
5.9 Exemples de fonctions triangulaires 141
5.10 Algorithme Monte-Carlo de MoCaNA 145
5.11 Diagramme de classes de MoCaNA 148

Table des définitions

Définition (Besoin en Fonds de Roulement) . . . 2
Définition (Agent) . . . 4

2.1 Définition (Équilibre walrasien) . . . 47
2.2 Définition (Équilibre général) . . . 50
2.3 Définition (Jeu stratégique) . . . 53
2.4 Définition (Jeu bayésien) . . . 54
2.5 Définition (Fonction d'utilité quasi-linéaire) . . . 54
2.6 Définition (Mécanisme) . . . 55
2.7 Définition (Mécanisme quasi-linéaire) . . . 55
2.8 Définition (Mécanisme déterministe) . . . 55
2.9 Définition (Mécanisme direct) . . . 55
2.10 Définition (Stratégie dominante) . . . 56
2.11 Définition (Mécanisme de Groves) . . . 56
2.12 Définition (Implémentation du bien être social) . . . 56
2.13 Définition (Mécanisme incitatif) . . . 56
2.14 Définition (Efficacité de Pareto) . . . 57
2.15 Définition (Rationalité individuelle) . . . 57
2.16 Définition (Mécanisme profitable) . . . 57
2.17 Définition (Mécanisme équilibré) . . . 57
2.18 Définition (Dictature) . . . 57
2.19 Définition (*Bargaining* de Nash) . . . 59

4.1 Définition (Jeu sous forme extensive) . . . 96
4.2 Définition (*Bargaining*) . . . 96
4.3 Définition (Agent incurieux) . . . 99
4.4 Définition (Agent strictement discret) . . . 99
4.5 Définition (Agent strictement curieux) . . . 99
4.6 Définition (Agent curieux et discret) . . . 99
4.7 Définition (Domaine d'acceptabilité) . . . 101
4.8 Définition (Domaine d'acceptabilité mutuel) . . . 102
4.9 Définition (Fonction d'utilité séparable) . . . 107

5.1 Définition (Loi normale multivariée) . . . 134
5.2 Définition (Fonction triangulaire) . . . 141

Table des théorèmes

2.1 Théorème (Incitation à l'honnêteté dans l'enchère de Vickrey [Vickrey, 1961]) 38
2.2 Théorème (Stabilité d'une coalition d'acheteurs dans l'enchère anglaise) 39
2.3 Théorème (Stabilité d'une coalition d'acheteurs dans l'enchère de Vickrey) 39
2.4 Théorème (Stabilité d'une coalition d'acheteurs dans l'enchère hollandaise) 39
2.5 Théorème (Existence d'un équilibre walrasien) 48
2.6 Théorème (Unicité de l'équilibre walrasien) 48
2.7 Théorème (Efficacité de Pareto) 48
2.8 Théorème (convergence du Tâtonnement) 50
2.9 Théorème (Existence et unicité de l'équilibre dans le processus de Negishi [Negishi, 1961]) 51
2.10 Théorème (Existence et unicité de l'équilibre du processus d'Edgeworth [Negishi, 1961]) 51
2.11 Théorème (Existence et unicité de l'équilibre du processus de Hahn [Hahn et Negishi, 1962]) 51
2.12 Théorème (Principe de révélation) 58
2.13 Théorème (Stabilité des mécanismes de Groves) 58
2.14 Théorème (Existence et unicité de la solution de Nash) 60
2.15 Théorème (Théorème de Myerson-Satterthwaite) 60
2.16 Théorème (Existence et unicité de la solution au problème du *bargaining* à information incomplète dans F^*) 61

4.1 Théorème (Utilité d'un agent strictement discret après un `reject`) 100
4.2 Théorème (Utilité d'un agent strictement curieux après un `reject`) 100
4.3 Théorème (Incitation partielle à l'honnêteté dans le cas mono-attribut) 104
4.5 Théorème (Incitation à rechercher un accord) 106
4.6 Théorème (Incitation partielle à l'honnêteté dans le cas multi-attributs) 109
4.7 Théorème (Incitation partielle à l'honnêteté de l'implémentation 1:n) 115

5.1 Théorème (Loi conditionnelle d'une loi normale multivariée) 135

Liste des tableaux

1.1 Tableau récapitulatif des places de marché multi-agents 19

2.1 Évaluations d'agents sur tous les lots de biens possibles 40
2.2 Table des utilités du jeu de sabotage 54
2.3 Utilité des agents en fonction du type de 2 et de son action 62
2.4 Utilité de l'agent 2 en fonction de la solution retenue 62

5.1 *Kernels* pour la régression de processus gaussiens et distance moyenne aux vraies propositions 139
5.2 Résultats de MoCaNA face aux finalistes de l'ANAC 2014 150
5.3 Résultats de MoCaNA avec élagage fixe face aux finalistes de l'ANAC 2014 151
5.4 Résultats de MoCaNA avec élagage variable face aux finalistes de l'ANAC 2014 152

Liste des Algorithmes

1 L'algorithme de tâtonnement 49
2 Protocole des deux millionnaires 114
3 MCTS 128
4 Apprentissage bayésien de l'utilité - cas numérique 142
5 Apprentissage bayésien de l'utilité - cas catégoriel 143

Introduction

1 Présentation du problème

On s'imagine rarement la complexité qui se cache derrière la fabrication des objets de notre quotidien. Pourtant de l'objet le plus simple comme le manuscrit que vous avez entre les mains à celui le plus compliqué comme l'ordinateur qui a servi à l'écrire, tous passent par les mêmes étapes : il faut d'abord extraire la matière première et la transformer plusieurs fois. Cette matière est ensuite utilisée pour produire des pièces, qui sont à leur tour assemblées. Après plusieurs étapes d'assemblage, les pièces assemblées constituent le produit final, qui doit souvent encore être conditionné avant d'être vendu.

Ces étapes sont rarement effectuées par la même entreprise. Le libéralisme pousse les entreprises à se spécialiser [1]. Pour produire, elles utilisent donc ce qui a été produit par d'autres. Lorsqu'on relie ces entreprises en fonction de leurs relation client/fournisseur, on obtient ce que l'on appelle une *supply chain* (ou *chaîne logistique*). Dans l'industrie en particulier, ces chaînes logistiques sont omniprésentes dans l'économie mondiale, et forment des filières rassemblant les différentes chaînes logistiques d'un même domaine et qui ont une cohésion propre. Cependant, cela ne signifie pas que tous les membres de cette chaîne logistique sont égaux. Les clients ont souvent l'avantage dans cette relation. Ils sont structurellement avantagés dans la *supply chain*. En effet, quand une entreprise en fournit une autre, elle émet également une facture. Cette dernière n'est cependant pas payée immédiatement. Elle est légalement payable sous 30 jours en France. Dans les faits, elle est bien souvent payée encore plus tard. Comme l'indique [Atradius, 2017], le délai moyen de paiement d'une facture en France en 2017 est de 42 jours. Ce décalage entre l'engagement des coûts de production et le remboursement par le client est appelé *besoin en fonds de roulement*.

Le besoin en fonds de roulement que subissent certaines entreprises peut avoir un effet dommageable pour l'économie dans son ensemble. Ainsi comme le souligne [REL, 2016], à l'échelle européenne, ce sont près de 981 milliards d'euros qui ont été perdus en 2015 en raison des mauvaises pratiques. Ce manque de trésorerie a plusieurs conséquences pour les entreprises qui en font les frais : il peut empêcher l'entreprise de croître, sa trésorerie étant immobilisée. Dans les cas les plus extrêmes ou en cas d'imprévu, ce manque de trésorerie peut même mener à un dépôt de bilan lorsque l'entreprise fait défaut, et cela même si les entreprises concernées ont des commandes en cours ou sont en attente de paiement d'une commande.

Il existe plusieurs définitions du besoin en fonds de roulement. On peut cependant en donner une définition imprécise mais suffisante dans le cadre de cette thèse :

1. Voir par exemple les travaux de David Ricardo (1772-1823) sur les avantages comparatifs.

Définition (Besoin en Fonds de Roulement)
Le Besoin en Fonds de Roulement (BFR) d'une entreprise est défini comme les actifs circulants (ses stocks et dettes d'exploitation) moins les créances générées par l'exploitation ; les dettes et créances financières, qui ne sont pas directement liées au fonds de roulement, ne sont pas comptabilisés :

$$BFR = Créances_{exploitation} + Stocks - Dettes_{exploitation}$$

Notons que l'augmentation du besoin en fonds de roulement d'une entreprise par la créance réduit celui de ses clients et toutes les entreprises dont elle est créancière (cette créance sera représentée chez ses clients comme une dette). On appelle *supply chain* l'ensemble formé par une entreprise ainsi que tous ses clients et fournisseurs, les fournisseurs de ses fournisseurs, les clients de ses clients, etc. La notion de *supply chain* est centrale dans l'appréhension du BFR, puisque c'est en collaborant au sein de la *supply chain* que les entreprises pourront abaisser leur BFR.

C'est sur ces problématiques qu'ont travaillé les membres du consortium RCSM (*Risk, Credit line and Supply chain Management*). Ce consortium s'est constitué dans le cadre d'un Fonds Unique Interministériel (FUI) et se compose de trois entreprises et quatre universités : Kyriba [2], Weave [3], Marsh [4], le LIP6 [5], ARMINES [6], l'INSA de Rouen [7] et l'IGR-IAE [8]. L'objectif principal des membres tient à optimiser le besoin en fonds de roulement des entreprises. Pour ce faire, le consortium a travaillé sur différents aspects de la question, passant en revue les modèles qui existaient déjà et construisant le sien. La concrétisation de ce travail consiste en différents rapports et écrits scientifiques, mais aussi en deux plates-formes logicielles :

- une plate-forme décisionnelle, permettant à une entreprise de définir les moyens d'optimiser son besoin en fonds de roulement en utilisant diverses méthodes,
- une plate-forme transactionnelle, permettant à une entreprise de revendre ses créances, ses bons de commande voire ses stocks, lui permettant ainsi de financer son BFR. C'est à certains aspects de cette plate-forme que cette thèse est consacrée.

L'architecture globale du projet est représentée sur la figure 1. On y retrouve les deux plates-formes, les différents acteurs, et l'information échangée.

On peut classer les méthodes de réduction du BFR selon deux critères : le type de méthode (investissement, logistique ou partage d'information) et son degré de collaboration (individuel ou collectif). La plate-forme transactionnelle est une plate-forme d'affacturage (*factoring* en anglais), qui fait partie des méthodes individuelles d'investissement (le fournisseur vend ses factures/bons de commande/stocks sur la plate-forme sans que cela ne nécessite aucune entente préalable avec un autre membre de la chaîne logistique). À titre d'exemple, il existe parmi les méthodes logistiques collectives ce que l'on appelle la Gestion Partagée des Approvisionnements, méthode consistant à mutualiser les transports logistiques d'approvisionnement en vue de réduire leur coût.

Les deux plates-formes du consortium sont reliées entre elles. En effet, la plate-forme décisionnelle peut utiliser la plate-forme transactionnelle comme moyen de réduire le BFR. De même, la plate-forme transactionnelle utilise un certain nombre d'informations qui lui sont fournies par la plate-forme décisionnelle. Ces interactions sont résumées dans la figure 1. Cette figure représente les différents acteurs (), les informations (), les types de financement () et enfin les plates-formes (et).

2. http://www.kyriba.fr
3. http://www.weave.eu
4. http://www.marsh.com
5. http://lip6.fr
6. http://armines.net
7. http://www.insa-rouen.fr
8. http://www.igr.univ-rennes1.fr

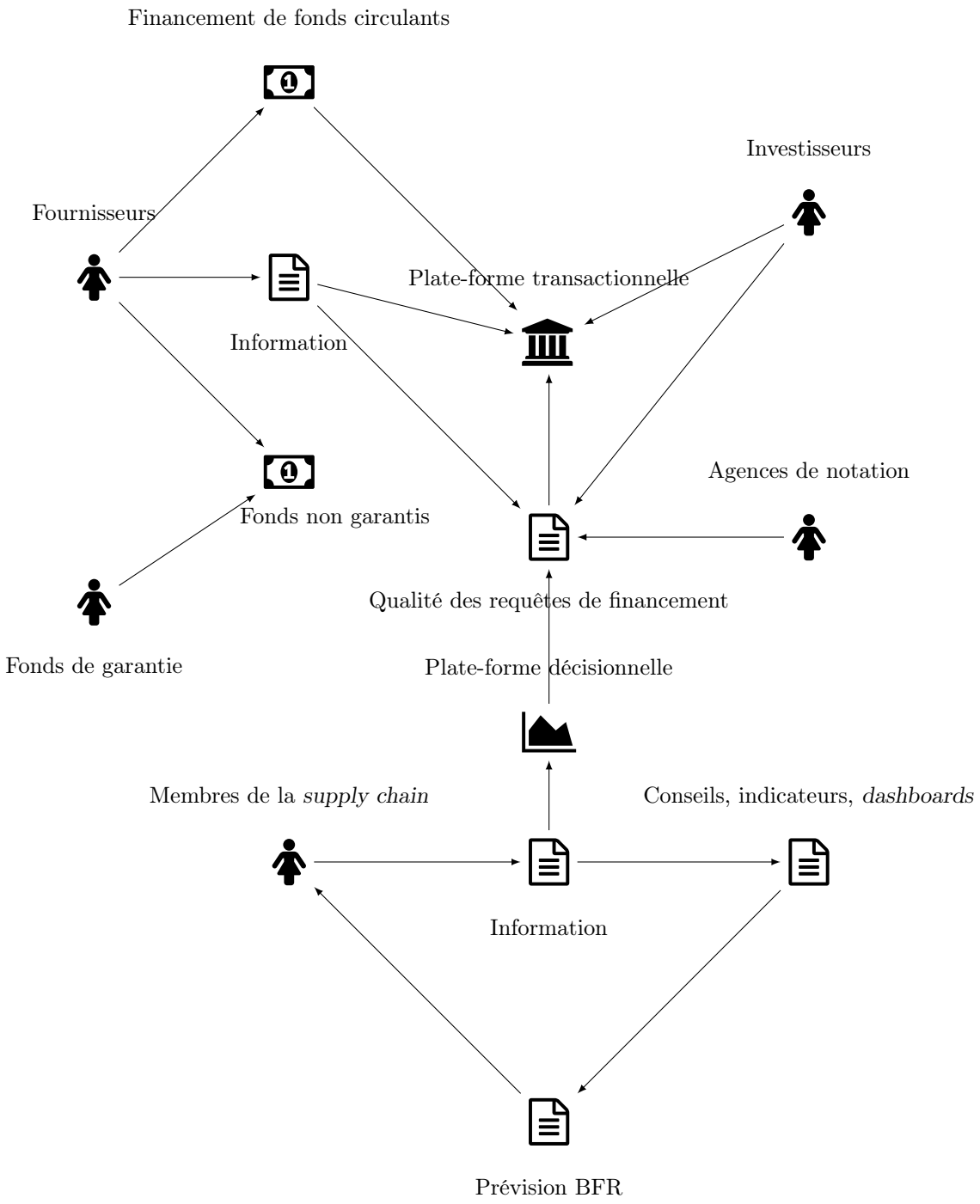


FIGURE 1 – Architecture du projet

2 Objet de cette thèse

Il existe déjà plusieurs places de marché permettant de vendre et acheter des factures [9]. Cependant, l'objectif de cette thèse est la conception d'une place de marché intelligente. Plusieurs aspects caractérisent notre place de marché. Tout d'abord, du fait des différents acteurs qui s'y connectent, il est nécessaire qu'elle soit aussi distribuée que possible. Ensuite, nous souhaitons que les acteurs de cette plate-forme puissent leur confier tout ou partie de leurs responsabilités de façon à ce que la plate-forme commercialise et achète des biens à leur place, ou bien qu'elle leur fasse des propositions auxquelles les acteurs se conforment ou non. Afin de représenter ces acteurs, nous proposons la conception d'agents logiciels autonomes, qui agiront en leur nom au sein de la place de marché. Enfin, et c'est la problématique principale que traitera cette thèse, les différents agents doivent être capables de communiquer et de s'organiser afin de négocier et d'arriver à un accord en suivant un protocole prédéfini.

Les *Systèmes Multi-Agents* (ou SMA) sont des méthodes répondant à ces différentes problématiques. Le domaine des Systèmes Multi-Agents est un sous-domaine de l'Intelligence Artificielle dans lequel on considère un ensemble d'entités intelligentes en interaction les unes avec les autres. Chacune de ces entités est appelée un *agent*.

Définition (Agent)
Un agent peut être défini comme une entité matérielle ou logicielle ayant les caractéristiques suivantes :

proactivité *Un agent peut agir sans qu'il lui soit donné aucun ordre ni aucune impulsion ;*

adaptation *Un agent est capable de prendre en compte les changements de son environnement et de réagir en conséquence ;*

aptitudes sociales *Un agent est capable de communiquer avec d'autres agents.*

La conception d'une plate forme de négociation pose plusieurs problèmes. Comme le soulignent [Baarslag *et al.*, 2015b], la négociation comprend trois parties importantes :

- l'objet ou le service qui est négocié,
- le protocole utilisé pour négocier,
- les agents de négociation.

L'objet de la négociation a déjà été spécifié : dans le cadre de cette thèse, il s'agit de négocier des créances. Nous allons donc nous focaliser sur les deux autres aspects : le protocole de négociation et les agents de négociation.

En ce qui concerne le protocole de négociation, nous nous basons sur le *bargaining*, afin que nos agents soient en mesure d'interagir directement les uns avec les autres. Les fournisseurs souhaitent en effet avoir le pouvoir de choisir avec quel investisseur ils négocient, et négocier les prix auprès de lui. Nous implémentons sur ce protocole un mécanisme permettant de contrer les agents curieux, agents entrant sur la plate-forme pour obtenir de l'information et non des biens, créant ainsi de nombreuses négociations inutiles qui encombrent la place de marché. Pour nous assurer que notre protocole de *bargaining* est bien résistant aux agents curieux, nous nous appuyons sur la théorie des jeux, et plus précisément sur le *mechanism design*.

C'est en utilisant le fait que la négociation est un jeu extensif que nous concevons notre agent. Nous explorons une famille de méthodes habituellement utilisée pour l'IA pour les jeux : les méthodes de Monte-Carlo, et plus précisément *Monte Carlo Tree Search* (MCTS). Cependant, le jeu de la négociation est un jeu particulier, et notre agent a besoin de modéliser ses adversaires pour jouer correctement. Nous adjoignons donc à MCTS des algorithmes d'apprentissage, qui permettent de modéliser l'utilité

9. The Receivable Exchange par exemple, qui a été racheté par LiquidX : `http://liquidx.com/`

et la stratégie de l'adversaire, respectivement la régression de processus gaussien et l'apprentissage bayésien.

L'ensemble des méthodes que nous utilisons pour notre place de marché intelligente est représenté à la figure 2.

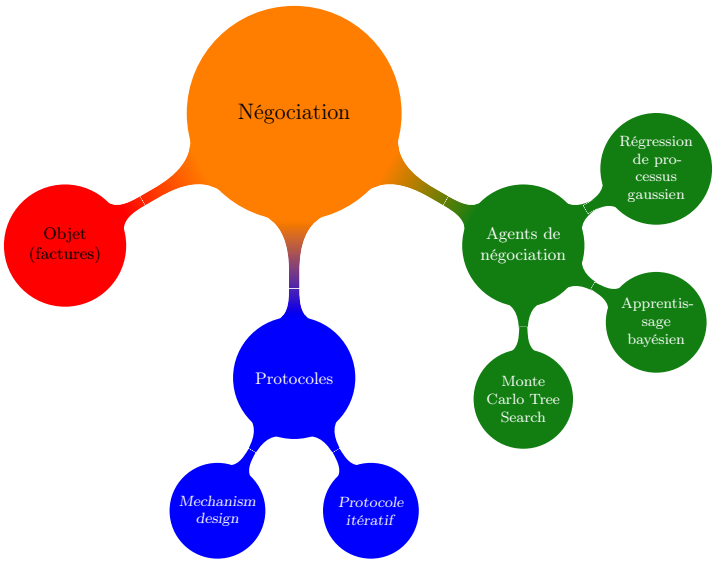


FIGURE 2 – Organisation des différents éléments de la négociation

3 Organisation de cette thèse

Dans le premier chapitre, nous réalisons une étude des places de marché de e-commerce en les classant selon le public auquel elles s'adressent, le protocole qu'elles utilisent, et la manière dont elles représentent l'utilité des agents. Nous décrivons ensuite notre propre plate-forme à travers une ontologie nommée MASK, et en précisons les spécificités techniques.

Nous nous proposons dans le second chapitre d'analyser les différents protocoles de négociation utilisés dans les places de marché de e-commerce multi-agents. Nous introduisons les concepts et les propriétés de base de ces protocoles ainsi que les variantes qui ont été conçues depuis. Nous introduisons dans un second temps le *mechanism design*, domaine mathématique ayant pour objet l'étude des mécanismes, qui s'appuie sur l'analyse mathématique des protocoles existants et propose d'en concevoir de nouveaux ayant des propriétés d'incitation à l'honnêteté.

Le troisième chapitre fait un état de l'art des agents existant dans la négociation automatique. Ces agents sont en général utilisés pour négocier dans le cadre du *bargaining*. En suivant le modèle BOA, nous présentons les différentes méthodes permettant de modéliser l'adversaire (son utilité, sa probabilité d'accepter une proposition, etc.). Nous présentons ensuite les différentes stratégies des agents négociants, et comment les techniques de modélisation y sont utilisées. Nous terminons par la présentation des stratégies d'acceptation de la proposition de l'adversaire utilisées dans ce contexte.

Dans le quatrième chapitre, nous proposons une manière de s'opposer aux agents curieux. Nous

commençons par donner une description formelle des agents curieux, puis proposons un protocole contrant naturellement ces agents. Nous l'analysons d'un point de vue expérimental et proposons une extension au contexte multilatéral. Nous étendons ensuite ce protocole du cas où la négociation porte sur un prix au cas où elle porte sur un nombre arbitraire d'attributs continus. Nous proposons ensuite trois implémentations : une centralisée, une décentralisée et une destinée à la négociation 1:n basée sur les enchères de Vickrey. Nous démontrons que les propriétés de notre protocole tiennent toujours dans ce cas.

Enfin, le dernier chapitre se concentre sur la création d'un agent dont la stratégie se base sur les méthodes de Monte-Carlo. Nous commençons par présenter l'algorithme MCTS, qui a eu beaucoup de succès dans les jeux extensifs. Nous expliquons ensuite en quoi la négociation peut être vue comme un cas très particulier de ces jeux. Nous proposons ensuite une stratégie basée sur les méthodes de Monte-Carlo et s'appuyant sur des méthodes d'*opponent modeling*, de *Gaussian Process Regression*, et d'apprentissage bayésien. Cette stratégie est moins contrainte que celles de l'état de l'art. En effet, elle supporte tous types d'attributs (continus comme discrets) et n'a pas besoin de se rapporter à une limite de la négociation en nombre de tours ou en temps. Nous analysons les performances de cet agent en face d'agents de l'*Automated Negotiating Agents Competition* 2014, et contre un agent aléatoire.

Nous concluons cette thèse en expliquant en quoi ces travaux constituent une avancée, d'un point de vue de la chaîne logistique comme d'un point de vue de la négociation automatique en proposant le traitement d'une nouvelle problématique intrinsèquement liée à l'essor de la négociation automatique (les comportements curieux), à la mise en évidence des liens entre négociation et jeux extensifs et à l'application de techniques qui en découlent.

Chapitre 1

Places de marché multi-agents

“
Consommez ! et les pelotes de ficelle de couleur tournaient comme des toupies, les feuilles de papier à fleurs battaient des ailes en enveloppant les achats pour en faire des petits paquets puis, en les groupant, des paquets moyens et, avec ceux-ci, de gros paquets, chacun d'eux ficelé avec un joli nœud.
Italo CALVINO, *Marcovaldo ou les saisons en ville*, trad. Roland Stragliati 1982
”

Sommaire

1.1 Introduction 8
1.2 Étude des places de marché 9
1.2.1 Clients des places de marché 9
1.2.2 Protocoles 16
1.2.3 Objet de la négociation 17
1.2.4 Préférences 17
1.2.5 Places de marché existantes 18
1.3 Notre plate-forme multi-agents pour le commerce de créances 22
1.3.1 OntoMASK 23
1.3.2 Considérations techniques 28
1.3.3 Discussion 31
1.4 Conclusion 32

1.1 Introduction

Notre manière d'acheter a énormément évolué au cours des deux dernières décennies. Elle a particulièrement été marquée par l'avènement d'un nouveau type de systèmes : le commerce électronique (ou e-commerce). Le commercce électronique utilise les possibilités offertes par Internet pour permettre aux clients d'acheter depuis chez eux. Mais au-delà du confort d'achat, le commerce électronique permet surtout aux vendeurs d'appliquer les avancées en informatique au commerce. Ces avancées concernent notamment l'intelligence artificielle, qui est utilisée pour aider et guider le consommateur. Il est même possible d'aller plus loin, en proposant au client de mettre sa confiance en la machine et la laisser acheter à sa place.

Le périmètre du e-commerce n'a cessé de s'étendre au cours des dernières années. Son histoire commence en 1979, année où Michael Aldrich développe le premier système d'achat-vente en ligne. En 1980, le minitel permet à des spécialistes de la vente par correspondance de développer leur service de e-commerce en France. Le e-commerce fait un véritable bond en avant en 1989 avec Compumarket, un système d'achat-vente qui permet de mettre des objets en vente. Ces objets sont répertoriés sur une base de données accessible par les acheteurs. Il est également possible d'utiliser une carte bancaire pour payer en ligne. Mais c'est la fin des années 1990 qui voit l'avènement du e-commerce tel que nous le connaissons aujourd'hui, avec la création d'Amazon, d'eBay et d'Alibaba. Ces systèmes, qui réussissent à survivre à la « bulle internet » ont durablement marqué le secteur. Après l'éclatement de la bulle, ils ont repris leur ascension pour devenir les systèmes innovants que l'on connaît aujourd'hui.

La croissance constante du chiffre d'affaires liée à ce domaine d'activité, qui est passé en France de 25 milliards d'euros en 2009 à 70 milliards d'euros en 2016 (selon la Fédération du commerce Électronique et de la Vente À Distance [1]) explique l'intérêt qu'il suscite auprès des grandes entreprises, et notamment des « GAFA » (Google Apple Facebook Amazon). Il semble en outre que l'intérêt des consommateurs pour ce nouveau marché tende à croître : les ménages français ont en effet vu leur nombre moyen d'achats en ligne annuels passer de 9,3 à 9,8 entre 2015 et 2016 (Kantar Worldpanel). En ce qui concerne le commerce entre entreprises, la tendance est la même : aux États-Unis, Forrester indique que l'e-commerce Business-to-Business atteint les 889 milliards de dollars, et représente 11% des ventes B2B totales. Selon la même étude, ce chiffre d'affaires devrait atteindre en 2021 1200 milliards de dollars, représentant 13.1% du marché.

Devant cette croissance, les grands acteurs du secteur, Amazon en tête, ont rapidement compris quels avantages ils pouvaient tirer des techniques d'intelligence artificielle dans le domaine, allant des systèmes de recommandation aux prédictions d'achat. Dans cet environnement, il paraît tout naturel qu'une littérature mettant en relation les systèmes multi-agents et l'e-commerce se soit développée, d'autant que ces deux sujets se prêtent particulièrement bien à cette interaction. En effet, la problématique du e-commerce est par essence multi-acteurs. Elle consiste essentiellement à aider les acheteurs à trouver le bien ou le service qui leur correspondra le mieux, et à aider les vendeurs à trouver les acheteurs qui sont intéressés par leurs prestations et à négocier les termes de l'accord entre ces deux acteurs.

Les systèmes multi-agents constituent un outil privilégié de mise en relation de l'économie et de l'informatique. Ils servent notamment à modéliser l'économie à différentes échelles, comme le montrent [Sylvain *et al.*, 2014, Goudet *et al.*, 2015, Tedeschi *et al.*, 2012], ou à automatiser les échanges, comme nous le montrons dans ce chapitre. Cela tient à la relation privilégiée qu'ont l'objet des systèmes multi-agents et celui de l'économie : l'interaction des agents. L'économie étudie un certain nombre

1. http://www.fevad.com/bilan-2016-e-commerce-france-cap-70-milliards-a-ete-franchi/

d'interactions particulière entre des agents comme des particuliers, des banques et des entreprises. Ces interactions peuvent être modélisées ou automatisées après formalisations par des agents autonomes.

Dans ce chapitre, nous présentons les systèmes multi-agents pour les places de marche d'e-commerce. Nous commençons par donner une introduction générale à ce domaine en utilisant et en étendant une taxonomie établie par [Kurbel et Loutchko, 2003], et donnons pour chaque modalité un tour d'horizon des principales plates-formes de chacun des concepts introduits. Nous donnons ensuite quelques éléments d'implémentation sur ces places de marché. Enfin, nous présentons la place de marché qui a fait l'objet de notre étude, Kinvo.

1.2 Étude des places de marché

Une classification des places de marchés a été établie par [Kurbel et Loutchko, 2003] selon plusieurs caractéristiques :

Les acteurs Une place de marché de e-commerce peut s'adresser à des professionnels (*businesses*) ou à des particuliers (*customers*). Ces acteurs peuvent jouer deux rôles : acheteurs ou vendeurs. Selon les acteurs, le fonctionnement de la place de marché est différent, car les contraintes (notamment légales) pesant sur une entreprise et celles pesant sur un particulier sont très différentes.

Le protocole Le protocole utilisé par les agents pour négocier se caractérise notamment par le nombre de vendeurs et acheteurs qu'il implique, ainsi que par le nombre d'échanges possibles pour arriver à un accord.

L'objet de la négociation Le domaine de négociation peut soit porter sur un unique attribut (le prix), soit sur plusieurs attributs comme la quantité de bien, leur qualité...

Les préférences Les préférences de l'agent peuvent s'exprimer de plusieurs manières : une distinction entre ce qui est acceptable et ce qui ne l'est pas, ou *via* une fonction d'utilité.

1.2.1 Clients des places de marché

Les acteurs les plus couramment considérés dans la littérature sur les places de marché sont les consommateurs individuels et les entreprises. On considère en général qu'un acteur n'interagit qu'avec un autre acteur à la fois (un acheteur faisant affaire avec un vendeur), et les places de marché ciblent généralement un type d'acteurs en priorité. Il y a donc théoriquement quatre possibilités en ce qui concerne les acheteurs et les vendeurs d'une place de marché [Turban *et al.*, 2015] : Business-to-Customer (ou B2C : une entreprise vend des produits à des particuliers), Customer-to-Business (ou C2B : un particulier propose d'acheter un produit sur mesure à une entreprise), Customer-to-Customer (ou C2C : des particuliers se vendent des produits) ou enfin Business-to-Business (ou B2B : l'acheteur et le vendeur sont des entreprises). Dans ce dernier cas, la vente correspond à la signature d'un contrat entre deux entreprises.

Ces quatre possibilités ne sont cependant pas également représentées. En effet, on trouve nombre de places de marché B2B permettant aux entreprises de commercer, et les sites de B2C où des entreprises vendent leurs biens sur une place de marché dédiée ne manquent pas. Les places de marché C2C commencent à apparaître elles aussi sous la forme de places de marché d'achat/vente d'occasion. Enfin, les places de marché C2B sont plus rares, et concernent des marchés bien particuliers (typiquement les places de marché de type « SmartGrids » pour le commerce de l'électricité). Le C2B peut aussi

désigner un modèle où les clients (des individuels) s'unissent et créent une demande. Cette demande est ensuite relayée par un tiers auprès d'entreprises ou de *freelances*. Ceux-ci proposent ensuite des enchères pour répondre à cette demande.

Il existe enfin des plates-formes qui ne rentrent pas dans ces catégories. Il peut s'agir de B2B2C (une entreprise faisant transiter des biens et services *via* une entreprise tierce), le Business-to-Employee (B2E) où les entreprises (ou le comité d'entreprise) fait profiter ses employés de biens ou services, ou encore le *collaborative commerce* où les acteurs cherchent à collaborer pour atteindre un objectif et l'*e-government* où une agence gouvernementale prend la place d'un des acteurs. Ces modèles n'ont cependant pas fait l'objet de modélisation multi-agents à notre connaissance.

Nous nous focalisons dans la suite de cette section sur les places de marché B2B, B2C et C2B (dans sa version où les consommateurs se rassemblent et formulent une demande à l'attention des entreprises du secteur). Les places de marché C2C n'ont pas encore fait l'objet d'une modélisation multi-agents.

1.2.1.1 Places de marché B2C

Un modèle de fonctionnement pour les places de marché B2C est proposé par [Guttman *et al.*, 1998]. Il est nommé *Customer Buying Behaviour*. Ce modèle divise l'acte d'achat en six étapes, décrites ci-dessous. La figure 1.1 donne une illustration de ce modèle.

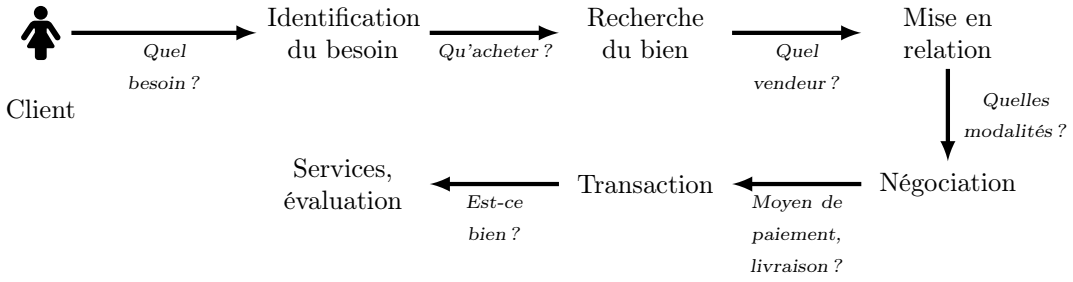


FIGURE 1.1 – Le modèle CBB, tel que décrit par [Guttman *et al.*, 1998]

Identification du besoin. À cette étape, le client rencontre un nouveau besoin, qui sera rempli en achetant un bien sur la place de marché. Bien que cette étape ne dépende pas essentiellement de la place de marché, il est possible de stimuler la demande en informant le client sur les produits et services proposés sur la place de marché. Aucune place de marché multi-agents ne propose un tel service, mais on en trouve dans nombre d'applications commerciales sous forme de systèmes de recommandation, comme Amazon, ou la place de marché de la FNAC.

Recherche du bien. Cette seconde étape consiste à identifier les biens qui correspondent au besoin du client sur la place de marché. La première étape ayant identifié le besoin, la seconde consiste à trouver le produit y répondant le mieux. Il peut exister plusieurs biens ou services semblables sans être identiques. Contrairement à l'identification du besoin, la place de marché peut intervenir à cette étape en évaluant précisément les besoins du client et en lui proposant les biens qui correspondent le mieux à sa requête.

Mise en relation. Cette troisième étape consiste à trouver les potentiels vendeurs proposant le bien choisi lors de l'étape précédente. Le choix du vendeur repose généralement sur le prix, mais ce n'est pas le seul critère (voir la section 1.2.3). On peut en effet ajouter dans les critères de choix la qualité de service, les modalités de livraison, les garanties, etc.

Négociation. Cette étape constitue le noyau central de la place de marché. Un descriptif plus détaillé des protocoles est donné dans le chapitre 2. Nous proposons de faire une distinction entre les protocoles selon le nombre d'agents qu'ils impliquent, tant du côté des acheteurs que des vendeurs. On peut trouver des protocoles bilatéraux (1 acheteur et 1 vendeur), des protocoles proposant un acheteur pour plusieurs vendeurs, des protocoles proposant un vendeur pour plusieurs acheteurs, et enfin des protocoles multilatéraux (plusieurs acheteurs négociant avec plusieurs vendeurs). Ces descriptions sont explicitées dans la section 1.2.2.

Transaction. Cette étape regroupe d'une part l'achat (choix du moyen de paiement, transfert de l'argent, etc.) et d'autre part la livraison du bien négocié (et potentiellement les modalités de livraison). Cette étape est principalement logistique et ne concerne pas spécifiquement la plate-forme. Pratiquement, cela peut cependant impliquer des contraintes légales du point de vue de la plate-forme. En effet, en France notamment, le fait de permettre le transfert de fonds *via* la plate-forme implique la création d'une entité particulière : un établissement de paiement.

Service et évaluation. Les différentes méthodes utilisées pour implémenter un système de recommandation pour une place de marché de e-commerce sont énumérés par [Jøsang *et al.*, 2007]. Le principe repose toujours d'abord sur la notation de chaque acteur par ceux qui ont fait affaire avec lui. La différence entre les différents systèmes va en fait reposer sur la manière d'utiliser ces données. Outre la simple somme et la moyenne, on trouve parmi ces méthode les statistiques bayésiennes, qui permettent à partir de jugements positifs ou négatifs de donner la probabilité que le jugement soit positif. Il est également possible de concevoir un système de réputation discret, reposant sur une classification de type « très fiable », « fiable », « neutre », « peu fiable » et « non fiable ». Ces systèmes reposent en général sur des tables de correspondance, et sur la fiabilité des agents qui notent. Ces derniers voient leurs notation modifiées selon qu'ils sur-notent ou sous-notent.

Les méthodes ci-dessus ne prennent pas en compte l'incertitude de la notation, et c'est à cette problématique que répondent les systèmes de croyances, comme celui que proposent [Yu et Singh, 2002]. Ces systèmes fonctionnent sur les croyances directes des agents connus. Ils reposent sur trois valeurs : l'incertitude, la croyance que l'agent noté est fiable, et la croyance qu'il n'est pas fiable, ces trois valeurs se sommant à 1. On obtient finalement des scores, chacun correspondant à la probabilité que l'agent soit dans dans une classe, mais ces probabilités ne se somment pas généralement à 1 : la différence à 1 correspond à l'incertitude sur l'agent.

Au-delà de la logique classique, [Sabater et Sierra, 2002] proposent d'utiliser la logique floue, afin de représenter l'agent noté aux classes « fiable » et « non fiable ». Dans cette optique, les agents donnant leur avis sur l'agent noté ont chacun une représentation floue.

Il est enfin possible de répercuter la notation de l'agent par les agents qui le connaissent directement vers les autres en pondérant à chaque fois par la confiance de chaque agent en ses connaissances, comme le montrent [Ziegler et Lausen, 2004]. Il est ainsi possible de créer un graphe qui représentera le système dans son ensemble et la confiance qu'a chaque agent en chacun de ses voisins.

1.2.1.2 Places de marché C2B

Dans les places de marché C2B, les clients se rassemblent pour créer une demande spécifique. Une fois cette demande créée, les entreprises peuvent y répondre, et entamer une négociation avec les clients. Sur la base de ce modèle, [Chen *et al.*, 2008] proposent un modèle nommé *Buyer Collective Purchasing* que nous reproduisons en le traduisant dans la figure 1.2. Du fait même de la manière particulière de procéder, ce modèle diffère des autres. Ici comme ailleurs cependant, certaines étapes (ici la dernière) ne concernent pas les agents intelligents.

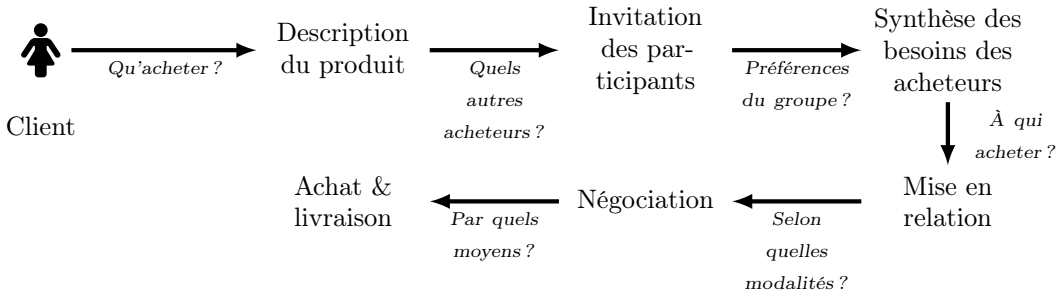


FIGURE 1.2 – Le modèle BCP, tel que décrit par [Chen *et al.*, 2008]

Description du produit. Durant cette étape, les clients expriment un besoin collectif, qui sera ultérieurement soumis aux potentiels vendeurs. Ce processus est introduit par un premier client créant une demande, permettant aux autres participants d'exprimer leurs besoins sur produit ou service demandé. Ces besoins sont ensuite agrégées sous la forme d'un arbre de critères. Chaque critère de niveau supérieur va se subdiviser en sous-critère au niveau inférieur, jusqu'à obtenir des critères atomiques. Cette tâche est dans la pratique accomplie par deux rôles : l'un est l'initiateur (l'agent qui exprime la demande), et l'autre est le coordinateur (l'agent qui transmet la demande). L'initiateur se chargera aussi de traduire les besoins des autres participants en arbre de critère.

Invitation des participants. Une fois la description du bien ou du service réalisée, une invitation est envoyée à chaque potentiel agent acheteur, autrement dit chaque agent ayant contacté le coordinateur. Ceux-ci le remplissent et transmettent leurs préférences à l'agent coordinateur.

Synthèse des besoins des acheteurs. Cette étape consiste à agréger les besoins des acheteurs afin d'en ressortir la description du bien ou du service qui sera négocié. Il existe de très nombreux mécanismes d'agrégation, le plus connu étant sans doute le vote. On peut ainsi imaginer que pour chaque choix, les différents acteurs votent pour leur attribut favori et que le choix global résultant constitue le bien négocié. Cela pose cependant un certain nombre de problèmes, le vote donnant lieu à des choix incohérents lorsque ceux-ci ne sont pas indépendants. Or, certains des choix à réaliser dans la description du produit sont liés. Prenons un exemple : une dizaine de clients cherchent à se coaliser pour acheter un ordinateur. On suppose qu'ils ont un choix à faire sur deux composants, la mémoire et le CPU. On suppose qu'il y a trois CPU au choix, un CPU faible (CPU A), un moyen (CPU B) et un fort (CPU C) ; et deux choix sur la mémoire, soit peu de mémoire (Mémoire 1), soit plus de mémoire (Mémoire 2). Les clients sont d'accord sur un point : la Mémoire 1 est trop faible pour les applications

nécessitant le CPU C. Cela entraînera une consommation d'électricité importante sans véritable gain de puissance. On suppose que les clients ont des desiderata différents :

- 4 veulent une machine puissante, et souhaitent donc acheter un CPU C avec une Mémoire 2
- 3 ont des besoins moins importants et préfèrent donc un CPU B avec une Mémoire 1
- 3 cherchent un ordinateur avec des capacités minimales, et préfèreraient un CPU A avec une Mémoire 1.

Dans ce cas, c'est le CPU C et la Mémoire 1 qui sont choisis, mécontentant tous les participants.

Ainsi, pour éviter ce genre de paradoxes, [Chen *et al.*, 2008] proposent d'utiliser un processus d'agrégation hiérarchique (*Analytic Hierarchic Process*, ou AHP ; voir [Saaty, 2013]). Cette méthode consiste à créer tout d'abord un but, qui se subdivise en critères, chacun ayant un poids. Ces critères se subdivisent eux-mêmes en sous-critères ayant chacun un poids. Au lieu de demander aux clients de donner leur avis sur les choix à faire, les acheteurs donnent l'importance qu'ils attribuent à chaque critère/sous-critère (les poids de l'arbre) ainsi que le score de chaque choix sur chaque critère non divisé (feuille de l'arbre n'ayant pas de sous-critère).

Reprenons l'exemple que nous avons donné précédemment. On suppose que les clients ont des besoins différents. Dans la première étape du processus, l'initiateur a construit un arbre de critères. On suppose que les clients considèrent qu'il y a deux critères d'évaluation pour un ordinateur : le critère écologique et le critère performance. En ce qui concerne le critère écologique, il se subdivise en deux sous-critères : l'impact carbone des matériaux utilisés et la consommation électrique. Le critère de performance se subdivise lui aussi en deux sous-critères : performance bureautique et performance développement. Ces critères et la manière dont ils se divisent forment un arbre.

Pour chaque nœud interne de l'arbre p, on considère qu'il est composé d'une somme pondérée du score sur chacun de ses fils. Chaque participant est invité à indiquer son estimation sur le poids sur p de chacun de ses n fils c_i. Le poids de l'ensemble des fils vaut 1, soit en notant w_{c_i} le poids du fils c_i :

$$\sum_{i \leq n} w_{c_i} = 1 \tag{1.1}$$

Nous obtenons dans notre exemple :

- une fonction objectif o divisée en deux critères p et e
- le critère performance p divisé en deux sous-critères p_b et p_d, qui sont des feuilles de l'arbre
- le critère écologique e divisé en deux sous-critères e_p et e_c, qui sont des feuilles de l'arbre

Les participants sont donc invités à donner une évaluation sur les coefficients w sachant que $w_e + w_p = 1$; $w_{e_p} + w_{e_c} = 1$ et $w_{p_d} + w_{p_b} = 1$.

On demande également à chaque participant de noter chaque alternative sur chaque critère ne se subdivisant pas, qui constituent les feuilles de l'arbre des critères. Les poids et les notes recueillis après la consultation des agents sont agrégés soit en utilisant un vote par approbation, soit en utilisant une moyenne, arithmétique ou géométrique. On calcule alors le score de chaque alternative en faisant la somme de ses notes pondérée par les poids de chaque coefficients. Le score de $A, 1$ est donc :

$$o(A,1) = w_e \cdot \left(w_{e_p} \cdot S_{e_p,A,1} + w_{e_c} \cdot S_{e_c,A,1}\right) + w_p \cdot \left(w_{p_b} \cdot S_{p_b,A,1} + w_{p_d} \cdot S_{p_d,A,1}\right) \tag{1.2}$$

et on choisit l'alternative maximisant la fonction o. On peut ainsi éviter les alternatives ne satisfaisant aucun critère des participants, comme cela était le cas avec un vote direct sur les caractéristiques du produit.

Mise en relation. Une fois les besoins des acheteurs agrégés, il est nécessaire de les mettre en relation avec des vendeurs capables de leur fournir les biens ou services demandés. Cette étape est

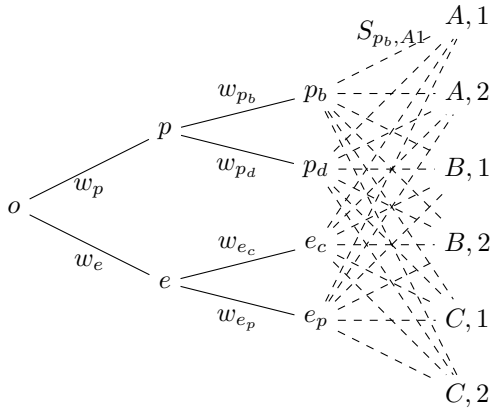


FIGURE 1.3 – Exemple d'*Analytic Hierarchic Tree*

identique à celles que l'on trouve dans les modèles CBB et BBT.

Négociation. La négociation va typiquement prendre en compte les différents critères du bien ou du service et les replacer dans l'arbre créé par l'AHP. Cette évaluation permet de départager les propositions faites par les vendeurs.

Achat & livraison. Comme pour les autres modèles, les agents n'ont pas grande influence dans cette étape. Cependant et contrairement aux autres modèles, le modèle BCP considère que la livraison et le paiement marquent la fin du processus : il n'y a pas d'évaluation ni de service après-vente. Cela tient en partie au fait que le bien ou le service est créé selon le besoin exprimé par les clients une unique fois, rassemblés en un groupe qui n'a pas de raison de perdurer. On peut malgré tout imaginer un système de réputation non sur les produits, mais sur la capacité du vendeur à respecter ses engagements dans un système où l'engagement se ferait au moment de l'offre des vendeurs.

1.2.1.3 Places de marché B2B

Le fonctionnement des places de marché e-commerce B2B diffère des places de marché e-commerce B2C et C2B dans leur fonctionnement. Cette différence est liée à la nature de ces types de marché : là où un achat se résume dans le B2C à une simple transaction, cela représente en fait un contrat inter-entreprises en B2B. Il est donc nécessaire d'utiliser un autre modèle. [He *et al.*, 2003] décrivent le modèle *Business-to-Business Transaction* (BBT), reproduit et traduit par nos soins en figure 1.4. Tout comme CBB, le modèle BBT n'implique pas l'utilisation d'agents intelligents à chacune de ses étapes. Plus particulièrement, des agents intelligents sont généralement employés pour les trois premières étapes. [He *et al.*, 2003] soulignent cependant qu'il est aussi possible d'en employer pour l'étape de constitution du contrat notamment en utilisant des agents spécialisés en droit.

Établissement d'un partenariat. Cette première étape est assez lourde à mettre en place : elle consiste à établir un partenariat entre plusieurs entreprises. Celles-ci peuvent se rassembler et constituer une « entreprise virtuelle » fournissant des produits complémentaires, ou en *supply chains*. La formation

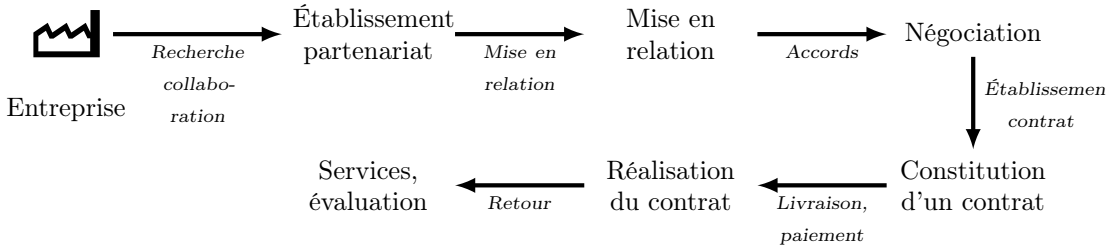


FIGURE 1.4 – Le modèle BBT, tel que décrit par [He *et al.*, 2003]

de ces partenariats est elle-même complexe. En effet, il est possible de créer des coalitions en utilisant soit des méthodes d'allocation de ressources ; ces méthodes sont en général utilisées par les entreprises virtuelles ; il est aussi possible d'utiliser des méthodes de planification comme le font les *supply chains*.

Mise en relation. Cette étape est identique à celle du modèle CBB. Une fois les partenariats contractés, l'entité (entreprise, entreprise virtuelle ou *supply chain*) recherche une entité à laquelle acheter ou vendre le bien ou le service concerné. On suppose que l'entreprise sait déjà quel produit ou service elle recherche et elle va naturellement rechercher l'entreprise qui répondra le mieux à ses besoins d'achat/de vente.

Négociation. Là encore, cette partie est identique à l'étape homonyme du modèle CBB. Il y a cependant une différence dans les protocoles de négociation utilisés dans les modèles B2C et dans les modèles B2B. Les modalités des protocoles utilisés dans les différents types de places de marché sont décrits dans la section 1.2.2.

Constitution d'un contrat. Cette étape consiste à formaliser le contrat tel qu'il a été négocié dans l'étape précédente. Elle est généralement opérée par des acteurs humains, mais il serait possible d'utiliser des agents spécialisés, capables de traduire en contrat légalement convenable les termes négociés. Il faut cependant noter que cette étape n'est pas si séparée de l'étape de négociation qu'elle ne le paraît. Les termes précis du contrat peuvent eux-même faire l'objet de négociation lors de sa rédaction, et il peut être difficile de déterminer qui est le plus à-même de les rédiger : agents de négociation ou agents spécialisés en droit.

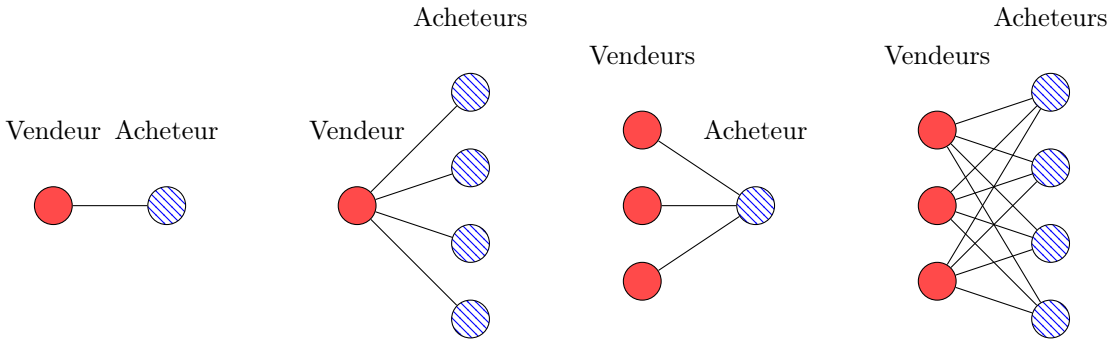
Réalisation du contrat. La réalisation du contrat comprend à la fois la réalisation de ce qui est spécifié dans le contrat par le vendeur, et le paiement de la compensation par l'acheteur. La complexité de cette étape et son aspect subjectif ne permettent pas aux agents logiciels d'y participer.

Service et évaluation. Cette étape est similaire à celle du modèle CBB. Il n'y a pas à notre connaissance de différence notable entre les systèmes permettant d'évaluer un vendeur selon que l'acheteur est ou non une entreprise.

1.2.2 Protocoles

Il s'agit ici de catégoriser les protocoles qui sont utilisés dans les différentes places de marché. Ce sujet étant particulièrement vaste, nous lui avons consacré le chapitre 2 de cette thèse. Néanmoins, nous donnons dans cette section un classement afin de faire correspondre chaque grand type de protocole aux types de place de marché précédemment mentionnée. Cette division, que nous appuyons sur [Kurbel et Loutchko, 2003] prend en compte le nombre d'agents engagés simultanément dans la négociation, parmi les acheteurs d'une part, et parmi les vendeurs d'autre part. Nous étendons cependant cette catégorisation afin d'y intégrer un second critère : le fait qu'un protocole soit itéré ou non.

La première caractéristique permettant de départager les protocoles est le nombre d'acheteurs et le nombre de vendeurs impliqués. Si un vendeur unique est en relation avec un acheteur unique à la fois pour chaque bien, le protocole est dit de type 1:1. Ce type de protocoles est assez peu présent sur les places de marché de e-commerce. En particulier, les négociations portant sur un bien n'ont pas lieu de manière simultanée mais séquentielle, ce qui peut être vécu comme une perte de temps. Un second type de protocoles consiste à faire négocier plusieurs acheteurs potentiels de manière simultanée avec un unique vendeur, le protocole permettant de rejeter les propositions des acheteurs n'ayant pas été retenues. On les appelle protocoles 1:n. Les enchères classiques entrent dans cette catégorie, et un certain nombre de places de marché mettent ce genre de protocoles en avant (eBay par exemple). Ce type de protocoles est très répandu et utilisé notamment dans les places de marché B2C et B2B. D'autres protocoles (typiquement utilisés dans le cadre du C2B, mais également parfois en B2B) permettent à plusieurs vendeurs de négocier de manière simultanée avec un unique acheteur, laissant le soin à ce dernier de choisir l'alternative qui lui semble la plus avantageuse et de rejeter les autres. Dans ce genre de protocoles, l'acheteur peut comparer les offres faites par plusieurs vendeurs et choisir celle qui lui correspond le mieux. Ces protocoles sont appelés protocoles m:1 (on représente classiquement les vendeurs par la lettre m et les acheteurs par la lettre n). Enfin, certains protocoles permettent de mettre en concurrence à la fois les acheteurs et les vendeurs. Dans ces protocoles, qualifiés de m:n, les acheteurs et les vendeurs négocient en même temps sur les mêmes types de biens ou de services avec plusieurs partenaires potentiels. Ce type de protocole est assez complexe, et n'a pas encore été mis en œuvre à notre connaissance dans les places de marché de e-commerce. Ces protocoles sont peu utilisés ; des exemples sont donnés dans la section 1.2.5. La figure 1.5 résume les différents types de protocoles selon le nombre d'acheteurs et de vendeurs impliqués.



(a) Protocole 1:1 (b) Protocole 1:n (c) Protocole m:1 (d) Protocole m:n

FIGURE 1.5 – Protocoles de négociation multi-agents

Un second paramètre à prendre en compte lorsqu'on s'intéresse aux protocoles de négociation est le nombre de tours permis pour parvenir à un accord. Il y a trois cas de figures :

1. Les protocoles à un tour sont des protocoles de type « à prendre ou à laisser » : les agents tenant un rôle (acheteur ou vendeur) font une proposition et les agents en face choisissent soit de l'accepter soit de la refuser. Certains types d'enchères, comme l'enchère de Vickrey, ou l'enchère scellée au premier prix figurent parmi ces protocoles.
2. Les protocoles ayant un nombre de tours illimités permettent d'échanger des propositions jusqu'à ce qu'un accord soit trouvé ou que les agents décident qu'ils ne peuvent pas trouver d'accord. Le problème principal de ces protocoles est que rien n'assure que la négociation se finira, que ce soit par un accord ou un désaccord. Les agents peuvent ainsi négocier sans jamais s'arrêter.
3. Les protocoles à limite de temps (ou de tours) sont souvent des reprises des précédents à cela près que si aucun accord n'est trouvé dans la limite impartie, la négociation est annulée et le bien ou le service n'est pas vendu.

En combinant cette classification avec la précédente, il est possible de classifier les protocoles de négociation, comme nous le faisons pour les protocoles des places de marché dans la section 1.2.5. La plupart des protocoles existent, sauf en ce qui concerne les protocoles m:n, moins étudiés que les autres, sans doute en raison de leur complexité.

1.2.3 Objet de la négociation

L'objet de la négociation constitue le troisième paramètre capital des places de marché multi-agents. Cet objet se caractérise d'abord par le nombre d'attributs sur lequel les négociations sont menées. Sur de nombreuses places de marché, seul le prix est négocié. Dans ce cas, on est dans le cadre d'une négociation mono-attribut. Il est cependant possible (voire courant dans les places de marché B2B ou C2B) de faire des négociations sur plusieurs attributs comme la qualité, la quantité, les délais de livraison, les pénalités de retard et les modalités de livraison. Ces paramètres peuvent être soit continus soit discrets. Le prix est un bon exemple d'attribut continu. D'un autre côté, il existe également des attributs discrets comme la couleur d'un bien, à choisir dans une gamme.

Le fait qu'une place de marché soit mono-attribut ou multi-attributs a une incidence assez importante. Dans le cas d'une place de marché mono-attribut, on négocie en général le prix. L'utilité d'un acheteur est décroissante selon le prix, l'utilité d'un vendeur, croissante. Dans le cas où seul le prix est négocié, pour un même bien, un acheteur désirera toujours acheter ce bien moins cher, et un vendeur le vendre plus cher. Une place de marché mono-attribut se résume donc pratiquement à un jeu à somme nulle. Ce qui est gagné par un agent est perdu par l'autre. Dans le cas où les agents négocient d'autres attributs, leur fonction d'utilité est généralement plus complexe, ce qui les conduit à des stratégies différentes. Dans ce contexte, il est notamment possible (et souhaitable) de rechercher des compromis qui satisfassent les différentes parties.

1.2.4 Préférences

Les plates-formes peuvent permettre aux agents d'exprimer des préférences sous forme binaire ou continue. Dans le cas où les préférences sont binaires, l'agent exprime à la plate-forme quelles sont les solutions acceptables pour lui. Cela signifie notamment que du point de vue de la plate-forme, deux solutions acceptables sont similaires. Ces places de marché sont en général utilisées dans un cadre coopératif où les agents négocient pour trouver une solution qui soit acceptable du point de vue de tous les agents.

Mais la majorité des places de marché sont utilisées pour de la négociation compétitive, où les agents cherchent à optimiser leur propre bien-être. Dans ce cas, les préférences de l'agent sont représentées dans la plate-forme sous forme d'une fonction d'utilité ou des ensembles flous portant sur les attributs de l'objet de négociation. Les fonctions d'utilité peuvent être de nombreuses formes. Dans le cadre des places de marché multi-agents que nous décrivons dans la section 1.2.5, les fonctions d'utilité se décomposent en une fonction pour chaque attribut. Une fois les valeurs obtenues pour chaque attribut, il est nécessaire d'agréger les préférences de ces attributs pour obtenir un *score* global. Dans les places de marché, ce score est représenté par une somme pondérée, mais il pourrait aussi l'être par des fonctions beaucoup plus complexes (voir notamment [Grabisch et Labreuche, 2010]).

1.2.5 Places de marché existantes

Dans la section précédente, nous avons listé les différents critères utilisés pour catégoriser les places de marché multi-agents. Dans cette section, nous classons les places de marché existantes selon ces critères. Nous donnons d'une part une description de ces places de marché et nous en résumons notre analyse dans le tableau 1.1.

TraCoNet [Sandholm, 1993] est une plate-forme de négociation basée sur le Contract Net de [Smith, 1980], avec l'ajout d'un message envoyé aux contractants perdants. Il permet à des véhicules de négocier l'emploi de routes. Cette plate-forme se compose de deux types d'agents. Les *managers* sont des agents disposant de la gestion des routes appelés centres de répartition et les *contractors* sont les agents cherchant à obtenir l'emploi de ces routes. Cette plate-forme est totalement décentralisée, et en dehors du protocole qui suppose une étape d'appel d'offre, il ne procède à aucune étape préalable.

Firefly [Shardanand et Maes, 1995] est un système de recommandation qui recherche les musiques qui correspondent le mieux aux préférences indiquées par l'utilisateur. Il ne s'agit pas à proprement parler d'une place de marché, puisque les agents ne peuvent pas acheter ou vendre dessus. Elle repose sur quatre algorithmes de recommandation.

Kasbah [Chavez et Maes, 1996] est une place de marché permettant d'acheter et vendre des biens. Chaque acteur, acheteur ou vendeur, confie ses préférences à un agent lui correspondant. Les vendeurs peuvent ainsi définir un bien à mettre en vente au moment où il le souhaite, en définissant un prix de réserve. De l'autre côté, l'acheteur peut demander à son agent de négocier en son nom pour un bien. La négociation est de type m:1 : un acheteur discute avec plusieurs vendeurs en même temps. Le protocole est cependant différent du protocole de [An *et al.*, 2016], puisqu'une acceptation est définitive. Les stratégies des agents négociateurs sont fixes.

Fishmarket [Di Napoli *et al.*, 1996, Rodríguez *et al.*, 1997] est une place de marché très proche d'AuctionBot, mais dont la structure est différente. Elle ne prend pas en compte tous les types d'enchères, mais uniquement l'enchère hollandaise, la plus utilisée pour la vente de poissons. Elle comporte quatre types d'agents : l'agent *auctioneer*, l'agent acheteur, l'agent vendeur et l'agent d'admission, qui décide de l'entrée des agents sur le marché.

MAGMA [Tsvetovatyy *et al.*, 1997] La *Minnesota AGent Marketplace Architecture* est une place de marché complète, comprenant des services divers. Elle embarque notamment des services bancaires (gestion des comptes, vérification de la disponibilité des fonds, transfert de fonds). On y trouve également un service de mise à disposition (*advertising*) sous la forme d'une liste des biens disponibles sur la place de marché (conceptuellement une base de donnée). MAGMA repose également sur une couche de communication. Cette couche utilise un format de sérialisation permettant d'échanger des informations (de paiement, sur les biens, pour négocier, ou encore

	PUBLIC			PROTOCOLE							NB ATTR.		PRÉFÉRENCES	
	B2C	**C2B**	**B2B**	1:1	1:n	m:1	m:n	**1 tr**	**∞ trs**	**n trs**	**1**	n	**dure**	**souple**
TraCoNet			•		•			•			•			
FireFly	•											•		•
Kasbah			•				•		•		•			•
Fishmarket			•		•				•		•			•
MAGMA	•			•	•			•	•		•			•
Jango			•									•		
AuctionBot	•				•	•		•	•		•			
Tete@Tete			•				•	•	•			•		•
MAgNET			•								•			•
eAuctionHouse/ Nomad	•		•		•			•	•		•			•
SilkRoad	•		•	•					•			•		•
GNP				•	•	•	•	•	•	•	•	•	•	•
Liang & Huang				•	•			•	•	•	•	•		•
INA		•				•			•			•	•	•

• tr : tour

TABLE 1.1 – Tableau récapitulatif des places de marché multi-agents

sur les objets achetés et vendus) en passant par un serveur central appelé *Relay Server*. Ce système est fonctionnel mais supporte uniquement une vingtaine d'utilisateurs. Enfin, l'entité centrale de MAGMA est l'agent commercial (*Trading Agent*). Celui-ci est relié à un service bancaire (accès aux services sus-décrits), à un inventaire contenant les objets achetés et les objets vendables, un manager de mise en vente connecté au service de mise à disposition et un service de négociation doté de deux modes : le mode manuel qui permet de faire de la négociation sous forme de *bargaining* et un mode dit « automatique » permettant en fait de simuler les négociations sous la forme d'une enchère de Vickrey. Le but de MAGMA est d'être ouverte, et si elle ne propose pas d'agent de négociation automatique au sens où nous l'entendons dans cette thèse (négociation de type *bargaining* automatisé), elle est en revanche assez ouverte pour qu'il soit possible de s'y connecter grâce à la couche de communication. La plate-forme elle-même est implémentée en Lisp, mais l'agent commercial est, lui, implémenté en Java.

Jango [Doorenbos *et al.*, 1997] est un agent utilisé pour parcourir des sites de e-commerce, et d'en extraire l'information. Il synthétise ensuite cette information. Il est utilisé dans le cadre de sites de e-commerce de type B2C. Il est notamment testé dans les mêmes conditions que BragingFinder (comparaisons de CD audio) ainsi que pour l'achat de logiciels. Il permet aux clients de passer moins de temps à rechercher un bien et de trouver de meilleurs prix. L'agent est accessible par une interface web, et comme SaaS (Software as a Service). Il embarque des techniques d'apprentissage automatique pour détecter de manière autonome les champs des différents sites à afficher.

AuctionBot [Wurman *et al.*, 1998] est une place de marché qui permet de négocier en ligne. Elle permet à des agents logiciels ou humains d'acheter en utilisant un système d'enchères. De nombreux systèmes d'enchères sont disponibles : enchères anglaises, hollandaises, premier prix scellé, Vickrey, Call ou CDA. Elle ne s'intéresse pas aux étapes qui précédent la négociation à proprement parler, et met simplement à disposition de l'acheteur un répertoire contenant les enchères en cours. Elle a été testée en cours, par des étudiants et en ligne à l'échelle d'une université.

Tete@Tete [Guttman et Maes, 1999] peut être vue comme une extension de Kasbah au cas multi-critère : la négociation ne porte plus uniquement sur le prix, mais sur d'autres caractéristiques du bien (il peut s'agir de la qualité du bien ou de toute autre caractéristique qui entre dans la négociation).

MAgNET [Dasgupta *et al.*, 1999] Cette place de marché est également implémentée en Java, et propose à l'acheteur un service proche de celui de Firefly, mais orienté B2B : la place de marché est conçue de manière à trouver les différents composants permettant de créer un produit. L'acheteur est représenté par un agent fixe, qui lorsque la requête est formulée par l'acheteur, génère un certain nombre d'agents mobiles qui interrogent les vendeurs sur leurs stocks. Ceux-ci sont également représentés par un agent fixe, qui après vérification des stocks/de la capacité de production, donnent une réponse à l'agent mobile. Celui-ci réalise alors une réservation, empêchant l'écoulement des stocks jusqu'à la prise de décision finale par l'acheteur. L'agent mobile revient ensuite à l'agent fixe, qui compare et synthétise les informations ramenées par les agents mobiles et constitue un plan. Il commande alors définitivement les composants dont il a besoin, annule les réservations sur les autres et paie des compensations financières pour chaque réservation annulée. La place de marché implémente deux améliorations : d'abord, lorsqu'un même vendeur doit être visité pour plusieurs biens, une unique demande sera faite, par un unique agent mobile. Ensuite, une amélioration a été implémentée pour la gestion de la *supply chain* : en effet, il est possible que plusieurs fournisseurs aient eux-mêmes un unique fournisseur, qui recevra alors plusieurs demandes qu'il refusera, alors qu'en définitive, un certain nombre de demandes seront ultimement annulées et un unique vendeur sera choisi. Pour éviter une telle situation, les demandes portent un identifiant qui les suit lors des demandes au sein de la supply chain. De cette façon, un fournisseur voyant deux commandes portant le même identifiant sait

qu'il n'aura en réalité qu'à produire un set de produits, même si deux demandes lui ont été adressées par plusieurs agents.

eAuctionHouse/Nomad [Sandholm et Huai, 2000, Huai et Sandholm, 1999] se divise en deux parties : une place de marché d'enchères intégrant plusieurs protocoles d'enchères : enchères simples (anglaise, hollandaise, premier prix et Vickrey), combinatoires et doubles. Le protocole est à choisir quand l'enchère est lancée. La place de marché est nommée eAuctionHouse. Il est possible d'acheter/vendre des lots de biens (différents ou identiques). Un système multi-agents l'utilise comme *middleware*, qui permet notamment de faire de la négociation automatique selon plusieurs types de stratégies, et de manière soit à emporter l'enchère, soit par curiosité, pour déduire le prix de réserve des autres acheteurs, ou au contraire pour générer du bruit et empêcher la récupération de ces informations. Ces agents sont en outre mobiles et peuvent migrer d'une place de marché à l'autre. La plate-forme offre également un support à la constitution de coalitions. Enfin, une interface graphique existe non seulement pour la place de marché, mais aussi pour la génération d'agents. Ces derniers peuvent donc être créés de manière graphique, à partir de briques. La création graphique donne lieu à la génération du code de l'agent et sa mise en ligne. Cette méthode permet notamment aux acteurs ne maîtrisant pas la programmation d'être mis sur un pied d'égalité avec les autres acteurs.

SilkRoad [Ströbel, 2000] est une plate-forme de négociation, dont les exemples se trouvent dans des contextes B2B et B2C. Elle intègre un protocole spécifique, bilatéral, et basé sur le protocole *Adjusted Winner* [Brams et Taylor, 1996]. Le but est de trouver un accord entre les deux partenaires qui les satisfasse au mieux. Le but n'est donc pas de mettre les deux agents négociant en compétition, mais de faire en sorte de contenter les négociateurs. Le protocole utilisé est proche de d'*Adjusted Winner*, mais pas identique. L'un de ses aspects est notamment une incitation à agir de manière honnête dans le protocole. L'article donne aussi deux applications : la réservation de billets d'avion et le choix d'une entreprise offrant des services de traductions.

GNP [Benyoucef *et al.*, 2000] La *Generic e-Negotiation Platform* de négociation n'est pas orientée agents ; elle repose sur une plate-forme permettant de modéliser tous types de jeux. Cette place de marché est dotée d'une extension pour la négociation ; tous les types de protocoles y sont modélisables, et la place de marché est accessible *via* un navigateur, en SaaS. La plate-forme accepte à la fois des agents autonomes et des agents humains. Elle expose une API pour y connecter des agents autonomes, mais ne propose pas d'agent générique. Enfin, seule la négociation (dans son aspect théorie des jeux) est pris en compte.

Plate-forme de Liang & Huang [Liang et Huang, 2000] Cette plate-forme est très complète. Elle intervient à plusieurs niveaux : d'abord la définition du type de contrat, du type de protocoles de négociation (six protocoles sont disponibles), elle embarque aussi la négociation à proprement parler, ainsi que la recherche de biens. La place de marché embarque des agents pour chacune de ces étapes, qui servent à faciliter et à automatiser ses différentes étapes. Elle se divise en plusieurs niveaux : le niveau de l'activité (typiquement la négociation), le niveau du contrat (typiquement les termes de la transaction) et le niveau du marché (en particulier, le choix du protocole).

INA [Shojaiemehr et Kuchaki Rafsanjani, 2013] se veut une implémentation du modèle BCP décrit en section 1.2.1.2 pour le e-commerce C2B. Elle implémente donc ses différentes étapes. Parmi les agents prévus à cet effet, on trouve le *group former*, un agent qui rassemble les utilisateurs qui ont des besoins similaires afin de créer un groupe. Un second agent récupère les plans créés par les acteurs, ou les récupère dans la base et l'envoie à un troisième agent, appelé agent de recherche. Celui-ci récupère l'ensemble des vendeurs susceptibles de vendre un bien ou un service faisant partie du plan. Les agents acheteurs contactent alors l'agent coordinateur extrait les préférences de chaque acheteur et les agrège et extrait une préférence globale, qui est à la fois

une préférence souple et un certain nombre de contraintes (préférences « dures »), demandant à chaque agent ce qu'il est prêt à accepter et quelles sont ses préférences. Pour ce faire, l'agent coordinateur utilise un *Analytic Hierarchy Process*, tel que décrit dans [Saaty, 2013]. Une fois les agents coordonnés, un agent négociateur procède à la négociation avec les différents vendeurs. Cette étape est réalisée selon un protocole particulier : un unique acheteur négocie sous la forme d'une enchère inversée, mais où l'agent acheteur fait aussi des recommandations en fonction des propositions qu'il a reçues par ailleurs.

Le tableau 1.1 rassemble les différentes places de marché décrites ci-dessus en prenant en compte les différents critères sus-décrits. La colonne PUBLIC rassemble le type d'acteurs auquel est adressée la place de marché. Ce public est parfois directement décrit par l'auteur, parfois non. Dans ce cas, nous nous appuyons sur les exemples donnés dans les articles ou à défaut sur le fonctionnement de la place de marché. La colonne PROTOCOLE rassemble deux paramètres du protocole : le nombre d'agents (acheteurs et vendeurs) concernés et le nombre de tours. Certaines places de marché proposent plusieurs protocoles. Dans ce cas, nous indiquons les paramètres de chaque protocole. La colonne ATTR. indique si la négociation est mono-attribut ou si elle est multi-attributs. Dans le cas où différents protocoles impliquent plusieurs possibilités, les deux cases sont cochées. Enfin, la dernière colonne indique si les préférences des agents sont des préférences selon une fonction d'utilité, si on recherche simplement le domaine pour lequel les deux agents sont prêts à faire affaire ou s'il s'agit d'une fonction d'utilité avec des contraintes supplémentaires. De nombreux cas de figure ont déjà été implémentés en ce qui concerne les places de marché B2B et B2C. Il n'existe en revanche qu'une unique implémentation en C2B. Au fur et à mesure du temps, les plates-formes ont embarqué de plus en plus de possibilités. En particulier, les plates-formes ont petit à petit incorporé plusieurs protocoles de négociation.

Certaines de ces plates-formes ne sont pas des places de marché à proprement parler, au sens où elles ne permettent pas aux agents de négocier. Nous les avons néanmoins incluses comme l'ont fait notamment [Guttman *et al.*, 1998], car elles font partie d'un écosystème, ou d'un référentiel qui sert de base à la comparaison de ces plates formes. C'est notamment le cas de Firefly et Jango. En dehors de ces cas particulier, les places de marché peuvent de diviser en deux catégories. Certaines sont très orientées multi-agents, comme Kasbah ou MAGMA, tandis que d'autres, comme celle proposée par [Liang et Huang, 2000] ou GNP, donnent en fait un cadre de négociation sur lequel des agents autonomes peuvent s'interfacer.

1.3 Notre plate-forme multi-agents pour le commerce de créances

Dans la section précédente, nous avons présenté les différentes places de marché existantes et leurs caractéristiques. Ces places de marché sont en général des prototypes et ont au mieux été utilisées à l'échelle d'une université. Dans cette section, nous introduisons Kinvo (pour *Kyriba Invoice*), une place de marché multi-agents industrielle utilisée dans le cadre du consortium RCSM, et dont le but est de faire de l'achat-vente de créance afin d'aider les acteurs économiques d'une *supply chain* à maîtriser leur BFR.

Les différentes parties de cette plate-forme constituent l'objet de cette thèse. Dans cette section, nous décrivons cette place de marché. Nous commençons par en décrire les différents concepts de la place de marché sous la forme d'une ontologie, en utilisant le formalisme OWL. Nous donnons ensuite un certain nombre d'éléments sur les technologies utilisées pour l'implémentation, ainsi que sur l'architecture applicative et d'implémentation de cette place de marché.

Décrivons maintenant notre propre plate-forme. Cette thèse consiste à implémenter un système de négociation automatique embarquant un système multi-agents et des agents autonomes capables de négocier en leur nom. Comme nous l'avons rappelé dans l'introduction, cette place de marché s'inscrit dans le contexte d'un projet plus large de réduction du besoin en fonds de roulement. Un certain nombre de propriétés de la plate-forme sont imposées par son utilisation.

Dans cette section, nous nous focalisons sur la place de marché dont l'automatisation est l'objet de cette thèse. Pour ce faire, nous présentons cette plate-forme sous la forme d'une ontologie (au format OWL). Nous abordons ensuite un certain nombre de considérations techniques.

1.3.1 OntoMASK

Nous présentons ici OntoMASK, l'ontologie de MASK, *Multi Agent Systems for Kinvo*, la plate-forme intelligente de RCSM. Cette ontologie nous permet de décrire à la fois la plate-forme et le noyau du système multi-agents qui l'utilise comme *middleware*.

La plate forme transactionnelle du projet *Risk, Credit line and Supply chain Management* a pour rôle de permettre à des fournisseurs de mettre leurs créances en vente, de façon à récupérer des liquidités (pour plus d'information sur la "Supply Chain Finance", se reporter à [Hofmann, 2005, Chen et Hu, 2011, Gupta et Dutta, 2011]). L'ontologie présentée dans cette section a pour ambition de présenter l'architecture et les concepts liés à cette plate-forme. Son organisation est modélisée en utilisant le format OWL (Web Ontology Language, *cf.* [Golbreich *et al.*, 2009]).

Nous présentons dans un premier temps ce qu'est une ontologie en substance, puis la méthodologie que nous avons utilisée pour construire OntoMASK. La troisième partie de cette section traite quant à elle de l'implémentation d'OntoMASK et présente les diverses sous-parties de l'ontologie, agrémentées d'une représentation graphique. Nous présentons également un graphe représentant l'architecture globale de l'ontologie puis concluons la section.

1.3.1.1 Utilisation d'OntoMASK

Avant de définir plus avant OntoMASK, il nous faut expliciter son utilisation. En effet, le rôle joué par une ontologie (modélisation, description, ou un mélange des deux) permettra de définir le niveau de formalisme attendu. Les ontologies remplissent deux rôles : raisonner sur ses concepts et inférer des relations entre les concepts qui y sont représentés d'une part et donner un aperçu sur les concepts d'un domaine les liens qu'ils ont entre eux d'autre part. OntoMASK est à mi-chemin. Il faut donc définir une ontologie rigoureuse et claire, qui permette de décrire précisément le fonctionnement et les entités de la plate-forme.

1.3.1.2 Méthodologie

Nous nous intéressons dans cette section à la méthodologie adoptée pour la conception de notre ontologie. Trois points de méthodologie sont abordés ci-dessous. Le premier concerne la recherche des informations. Le second traite de la façon dont nous avons procédé pour structurer les données, afin d'obtenir un « squelette » d'ontologie. Le troisième point concerne enfin l'implémentation concrète d'OntoMASK et de sa formalisation. Notons qu'un point manque, il s'agit du test et des raisonnements

faits sur OntoMASK. Faute de données concrètes en entrée, cette étape n'a pas pu être réalisée mais elle reste l'une des perspectives à court terme de ce travail.

Recherche des informations. On distingue généralement trois types de méthodologies pour la définition des classes d'une ontologie. La première est dite « *top-down* » et consiste à partir des concepts de haut niveau, et à les raffiner. La seconde méthodologie commune de recherche d'information est dite « *bottom-up* » et consiste à partir des concecpts les plus concrets, et à les abstraire afin d'en déduire des concepts de plus haut niveau. Enfin, la méthodologie « *middle-out* » part de concepts connus, et les raffine ou les abstrait quand cela est nécessaire.

Les connaissances nécessaires à la création d'OntoMASK ont plusieurs sources. D'une part, nous avons recherché dans un certain nombre de documents techniques définissant la logique fonctionnelle et l'implémentation technique de Kinvo. Ensuite, nous nous sommes appuyés sur un certain nombre d'articles traitant de places de marché multi-agents décrites dans la section 1.2.5 afin de définir les différentes parties de notre architecture. Ces éléments mêlent à la fois des concepts haut niveau et des concepts plus concrets. Selon les cas, il nous a parfois fallu raffiner certains concepts et parfois en rassembler certains en classes plus générales. La plupart du temps, il a été nécessaire de faire les deux. Notre méthodologie peut de ce point de vue être catégorisée comme "middle-out".

Structuration des données. Une fois les données collectées, nous avons entrepris de les structurer. Pour cela, il nous a fallu définir les relations qu'elles avaient entre elles. Nous avons ainsi pu construire un squelette de modèle, dans lequel nous avons placé les principaux concepts et les liens qui les unissent (généralisation ou lien sémantique). Ce modèle n'est pas aussi formel que l'implémentation ci-dessous mais c'est sur ce modèle que la formalisation OWL sera implémentée.

Formalisation d'OntoMASK. La dernière étape de notre démarche a consisté à formaliser le modèle obtenu à l'étape précédente. Nous avons pour cela utilisé un outil dédié à la création d'ontologies en OWL, Protégé [Noy *et al.*, 2000]. Cet outil nous a permis de passer d'un squelette de modèle à un modèle formel et précis respectant le standard OWL. La définition des différentes parties d'OntoMASK sont décrites dans la section suivante.

1.3.1.3 Implémentation

Cette section définit l'implémentation concrète d'OntoMASK en format OWL sous la forme de graphes produits par Protégé. Nous avons aussi divisé ce graphe en plusieurs sous-graphes, afin de pouvoir la visualiser plus facilement. Nous définissons d'abord les classes représentant les acteurs physiques de la plate-forme. La seconde sous-section décrit les agents de la plate-forme. La dernière catégorie de classes est constituée des biens qui pourront être échangés à terme sur la plate-forme. Les différentes figures ci-dessous ont été créées à l'aide du module d'export de Protégé.

Acteurs. La figure 1.7 présente les différents acteurs physiques de la plate-forme. Ces acteurs sont séparés en deux catégories. D'une part, les entreprises, qui utilisent la plate-forme et de l'autre, le `Program_assistant`, le gestionnaire de la plate-forme, tiers de confiance.

Les entreprises sont les utilisateurs de la plate-forme, parmi lesquels le `Cédant` a fourni un bien au `Donneur_d_ordre`. La facture, le bon de commande ou le stock qui en résultent forment les biens

échangés sur la plate-forme (*cf.* section 1.3.1.3). L'`Acheteur` est quant à lui un investisseur (un *factor*) souhaitant acheter les créances du `Cédant` contre une ristourne, générant ainsi un profit pour l'`Acheteur`.

Les acteurs de notre plate-forme sont représentés dans le système par des agents intelligents (*cf.* section 1.3.1.3), auxquels ils peuvent donner des ordres ou exprimer des préférences. Les agents agiront ensuite pour leur compte, leur permettant d'acheter, et de vendre des biens.

L'acteur `Donneur_d_ordre` est particulier. Il n'a aucun rôle dans notre architecture. Selon l'évolution de la plate-forme, nous espérons cependant être en mesure de lui permettre de « certifier » qu'il paiera un bien à l'échéance indiquée et ainsi améliorer la garantie sur ce bien.

Agents. La figure 1.6 donne un aperçu du graphe des agents et de leur liens hiérarchiques et sémantiques. Les agents héritent tous de la classe *Agent*.

Les agents de notre plate-forme servent à la négociation des lots. Ils héritent de `Agent_negociateur`. L'acheteur et le cédant ne conduisent eux-même pas les négociations sur les lots. Ils créent à cette fin un `Agent_negociateur` qui utilisera des stratégies afin de négocier avec l'agent partenaire. Ces stratégies sont décrites dans le chapitre 5.

Bien. Cette hiérarchie, représentée en figure 1.8 s'intéresse aux échangeables. Parmi eux se trouve l'unité de base, le `Bien`, qui peut être de trois types :

- il peut s'agir d'une facture non encore honorée et que le cédant souhaite financer jusqu'à son échéance ;
- il peut aussi s'agir d'un bon de commande. Le cédant peut avoir besoin de liquidité avant de produire son bien ou son service. Il cherchera dans ce cas à faire acheter son bon de commande par un investisseur afin d'obtenir les liquidités qui lui manquent ;
- il peut enfin s'agir de stocks de matière première voire de produits finis que le cédant veut faire financer pour les mêmes raisons qu'un bon de commande ou une facture.

L'implémentation concrète de la plate-forme ne concerne que les factures, mais la plate-forme est prévue pour pouvoir s'étendre à ces deux autres types de bien.

Un `Lot` est composé de plusieurs biens, potentiellement différents. Une fois le `Lot` constitué, il est mis en vente, et les acheteurs peuvent faire des offres dessus. Une offre est représentée sous la forme d'une `Offre` ; cette classe représente aussi bien les offres et les contre-offres.

Un `Echangeable` est soit un `Lot` soit un `Bien` et est caractérisé par son `Statut`, qui peut être `Prive`, c'est à dire que l'`Echangeable` n'a pas été mis en vente et visible uniquement par le `Cédant` qui le possède. Il peut aussi être `À_vendre` ou `Vendu`.

Aperçu. La figure 1.9 donne un résumé de l'architecture de la plate-forme transactionnelle. Elle a été créée avec le module d'export de Protégé.

Ces figures permettent notamment de voir les relations qui existent entre les hiérarchies définies ci-dessus qui ne sont pas représentées dans les figures précédentes.

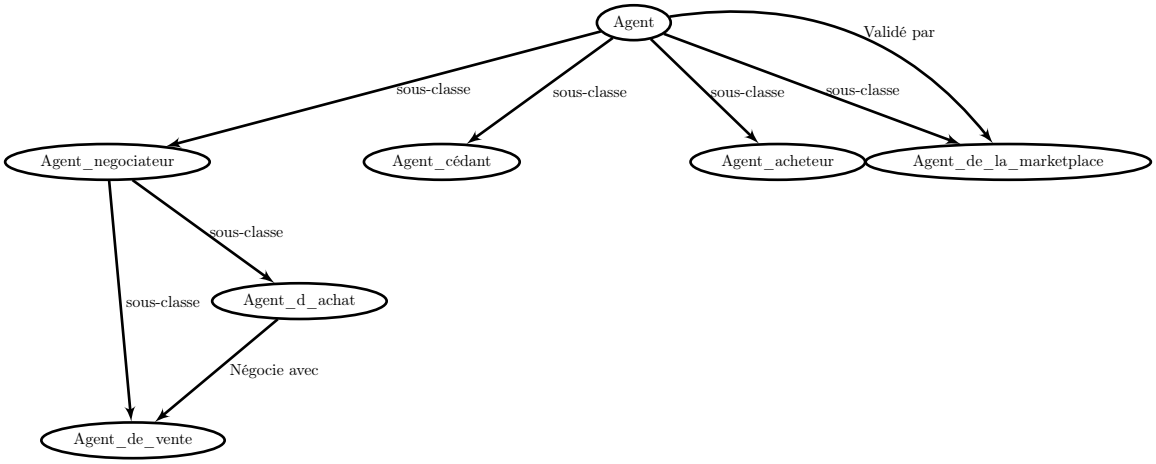


FIGURE 1.6 – Hiérarchie des agents

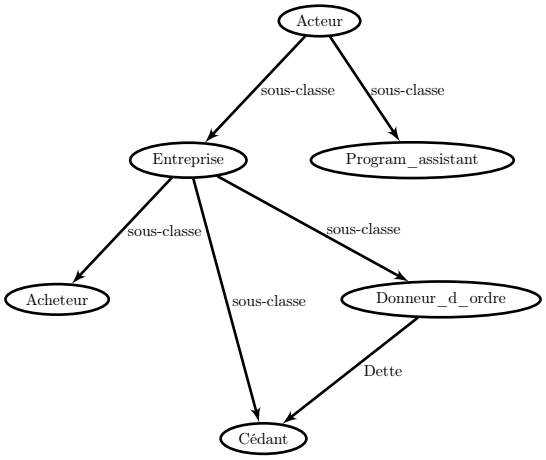


FIGURE 1.7 – Hiérarchie des acteurs

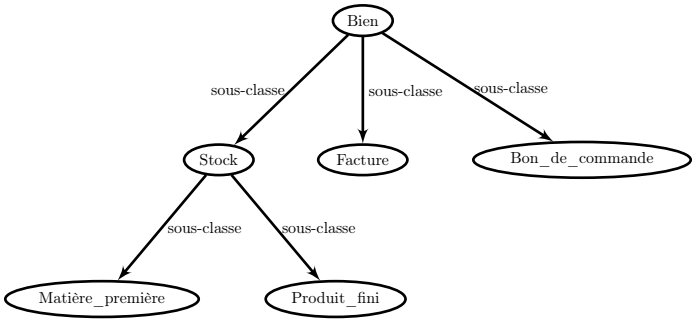


FIGURE 1.8 – Hiérarchie des biens

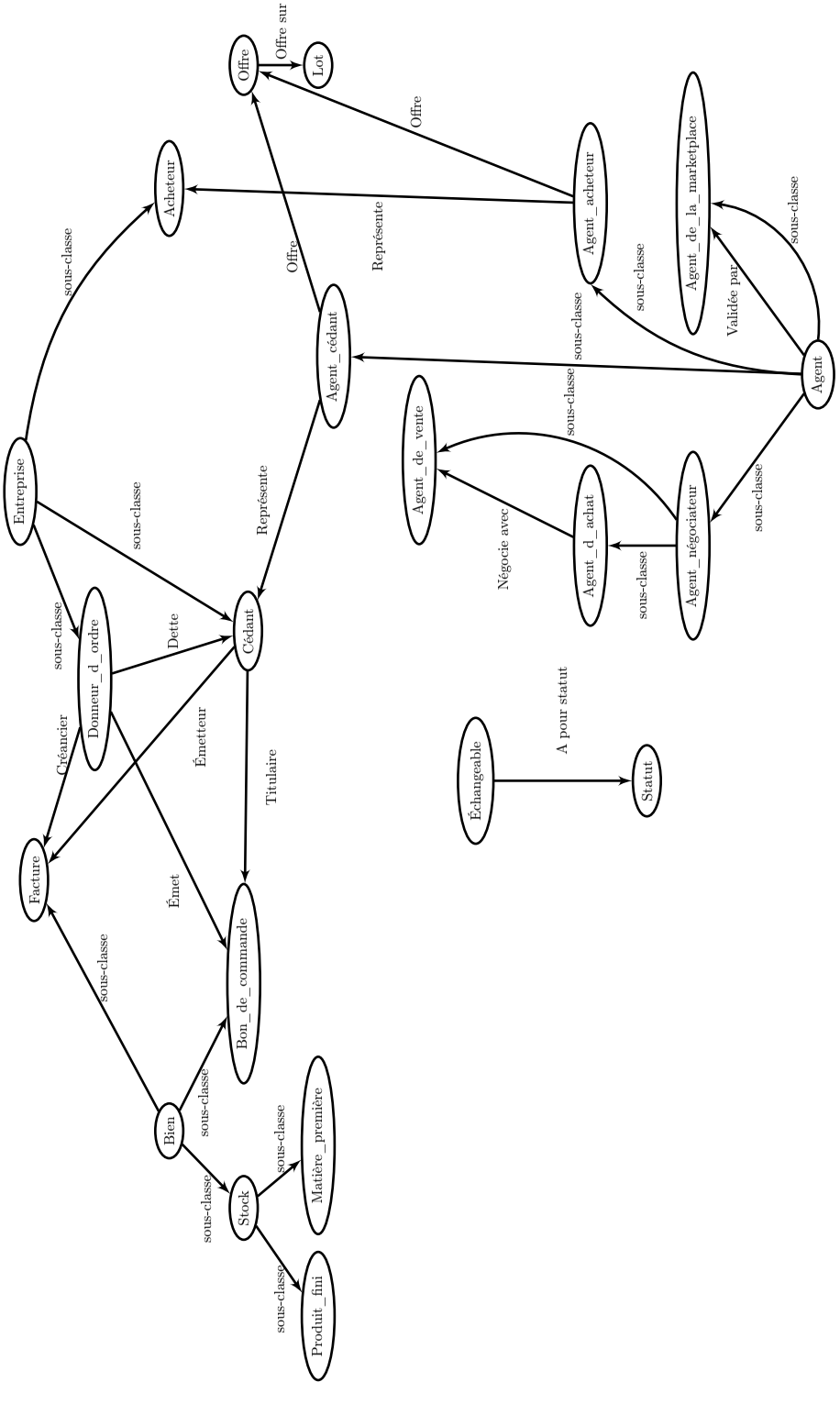


FIGURE 1.9 – Ontologie de la plate-forme

1.3.2 Considérations techniques

Cette sous-section donne un bref résumé des tenants et des aboutissants de l'architecture technique de la place de marché qui fait l'objet de cette thèse. Elle est reliée à Kyriba App, une application en SaaS (Software as a Service) divisée en modules, ayant chacun pour objet un aspect différent de la trésorerie.

1.3.2.1 Kyriba App

Avant de définir l'architecture de la place de marché, il est nécessaire de donner un apperçu de l'architecture applicative de Kyriba, car la place de marché est déployée conjointement à celle-ci, et repose entre autres sur les services qu'elle offre.

La Figure 1.10 donne un aperçu de l'architecture applicative de Kyriba. L'application est déployée sur plusieurs serveurs. La connexion à l'application se fait *via* un navigateur Internet. Les ERP et autres applications des clients et banques peuvent aussi se connecter. L'application elle-même se divise en deux zones, protégées par des pare-feux.

Ces deux zones contiennent plusieurs serveurs.

La zone Front contient :

- le serveur Front (webserver), qui est en charge de l'affichage et la gestion des contrôles présents sur la page affichée par le navigateur. Il permet donc d'accéder aux fonctionnalités présentes dans le serveur Back, qui n'est pour sa part pas accessible *via* Internet.
- Le serveur d'authentification qui permet la sécurité de l'application.

La zone Back contient pour sa part :

- le serveur Back (JavaEE application server), qui gère toute la logique métier, les calculs compliqués, la communication et la gestion de la session.
- Une base de donnée (Oracle SQL) « cloisonnée », c'est à dire que les données des clients ne sont accessibles que par eux.
- Un système de fichier partiellement cloisonné (les logs sont aussi enregistrés ici, accessibles uniquement par les administrateurs de Kyriba).

Architecture de déploiement La Figure 1.11 donne la description de l'architecture de déploiement de Kyriba. On y reconnaît l'architecture applicative (simplifiée). Kyriba est un logiciel en SaaS, mais il est réparti entre plusieurs plates-formes indépendantes. Comme on peut le voir, chaque plate-forme a sa propre base de données, ses propres zones front et back, et est donc techniquement indépendante.

1.3.2.2 Technologies utilisées

Kyriba est implémenté en Java et tourne sur des serveurs Tomcat. La liaison entre les serveurs back et front est réalisée au moyen de Enterprise Java Bean. La gestion de la persistence est elle gérée par EclipseLink, un framework de persistence certifié JEE.

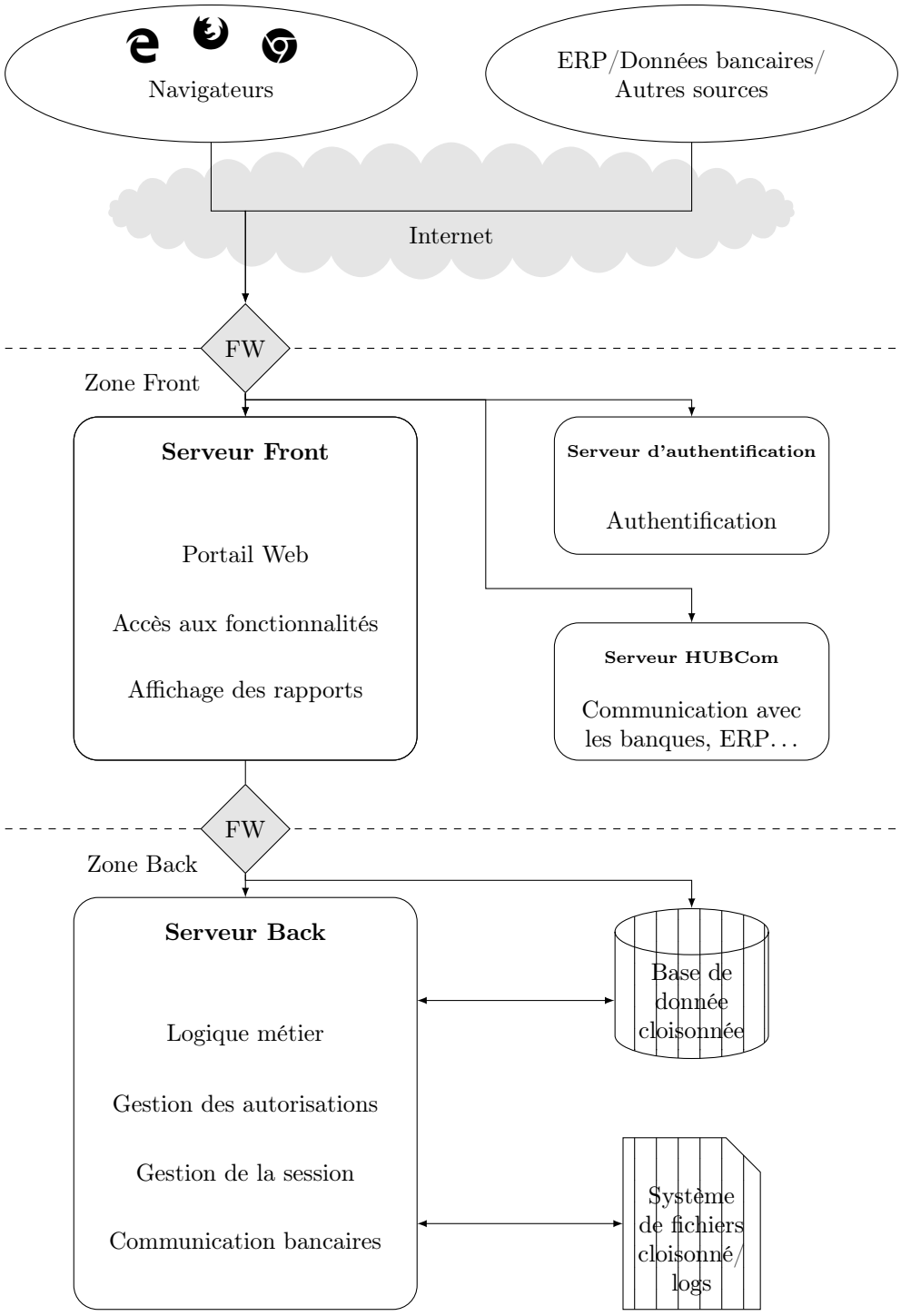


FIGURE 1.10 – Architecture applicative d'une plate-forme Kyriba App

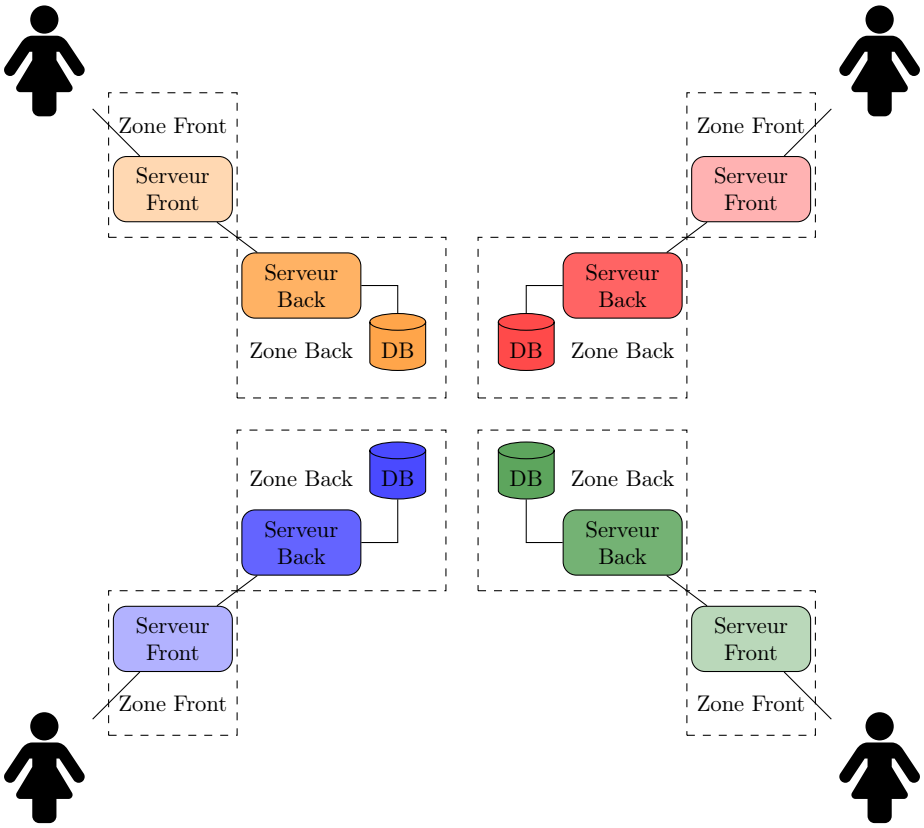


FIGURE 1.11 – Architecture physique et organisation des différentes plates-formes

1.3.2.3 Architecture de la plate-forme

L'architecture de Kyriba ayant été précisée, il est désormais possible de décrire l'architecture de la plate-forme.

La Figure 1.12 représente l'architecture fonctionnelle de la plate-forme. Celle-ci s'appuie sur Kyriba et l'utilise comme bibliothèque logicielle.

La plate-forme est divisée en plusieurs modules. A l'instar de Kyriba, elle s'appuie sur un ensemble de mécanismes, regroupés dans le module services techniques. Un second module (*Data Transfer Object*), lui aussi technique, permet la communication entre les différentes instances. Toute interaction entre les instances passe forcément par ce module.

Sur ces services techniques est construite la logique métier. Le module `kin-common` est le module métier contenant ce qui est commun à tous les clients, investisseurs comme fournisseurs. Trois modules héritent et étendent celui-ci. Deux de ces trois modules sont ceux de clients particuliers, l'un pour l'investisseur (*purchaser*) et l'autre pour le cédant (*seller*). Enfin, un package est réservé au *program assistant*, rôle de l'utilisateur système qui gère la place de marché.

Notons que du point de vue de Kyriba, Kinvo est un module supplémentaire, qui s'appuie lui aussi sur ses modules communs.

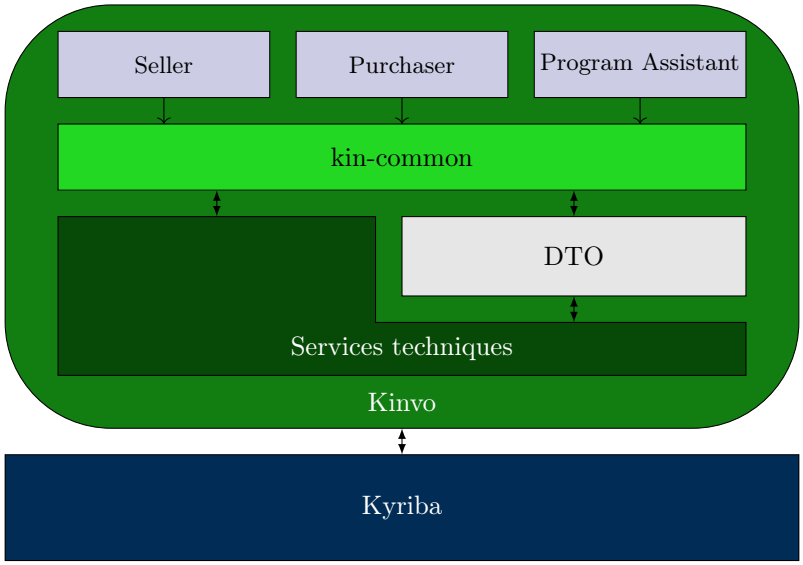


FIGURE 1.12 – Architecture fonctionnelle et organisation des modules

Architecture de déploiement Comme le montre la Figure 1.13, Kinvo apporte un changement assez important à l'architecture de déploiement de Kyriba. En effet, afin que les différentes instances puissent communiquer de façon sécurisée *via* Internet, il est nécessaire de relier les éléments par un Virtual Private Network (VPN).

D'un point de vue technologique, il n'y a pas de différence majeure entre Kyriba et Kinvo. Le module de communication de la plate-forme qui permet aux différentes instances de communiquer entre elles utilise lui-même les Enterprise Java Beans, mais dans le contexte du VPN.

1.3.3 Discussion

Dans cette section, nous avons présenté une ontologie servant de modèle à notre plate-forme multi-agents pour le commerce de factures, de bons de commande et de stocks. Nous nous sommes pour cela appuyés sur les architectures décrites dans la littérature. Nous avons également montré quelle méthodologie nous avons adoptée afin de concevoir cette ontologie, ainsi que les outils et formats que nous avons utilisés pour son implémentation. Enfin, nous avons décrit les différentes parties d'OntoMASK, et la façon dont elles sont reliées entre elles.

Notre ontologie, OntoMASK a été conçue comme modélisation de la plate-forme que nous implémentons, et nous avons pris cet élément en compte lors de sa conception. Nous avons notamment dû répondre à un double critère de rigueur et de simplicité.

Nous avons également explicité les composantes techniques de cette plate-forme. Celle-ci permet à des acteurs présents sur plusieurs serveurs de négocier malgré tout les uns avec les autres. La place de marché, implémentée en Java, bénéficie de l'architecture fournie par Kyriba App et des services proposés par ses bibliothèques. Elle reste cependant spécifique, tant dans le fait qu'elle permet aux acteurs de négocier depuis différents serveurs que parce qu'elle implémente ses propres fonctionnalités.

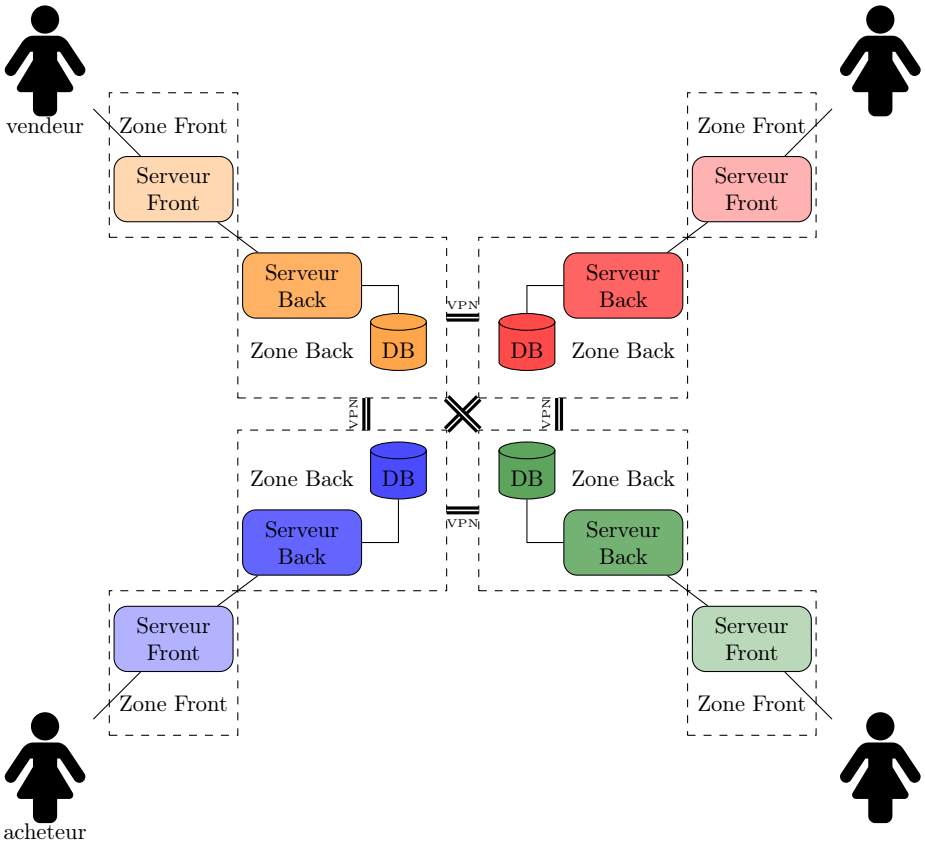


FIGURE 1.13 – Architecture physique et modifications amenées par la plate-forme

1.4 Conclusion

La plate-forme que nous avons automatisé a pour objectif la réduction du besoin en fonds de roulement. Il s'agit d'une place de marché utilisable par les entreprises pour revendre leurs créances, stocks ou bons de commande et qui leur servira à financer leur BFR auprès d'investisseurs (banques ou fonds). Nous sommes donc dans un contexte particulier : d'une part, il s'agit de commerce *B2B*, d'autre part, les acteurs présents sur la plate-forme en font une place de marché très à part. Bien que la plate-forme propose du *B2B*, les moyens des acheteurs et ceux des vendeurs sont complètement différentes : d'un côté, les acheteurs sont des banques ou des fonds d'investissements dotés d'importants fonds. De l'autre, la cible de la plate-forme est constituée de PME, parmi lesquelles certaines viennent parce qu'il leur est difficile d'emprunter auprès de banques. Cette asymétrie de moyen doit être prise en considération : il ne faut pas qu'elle permette aux acheteurs d'abuser de leur supériorité afin de tirer des profits aux dépens des vendeurs. Le but est qu'elle soit profitable à tous. Ces aspects ont nécessairement un impact important sur le protocole de négociation, qui est discuté au chapitre 4, ainsi que sur l'agent de négociation, exposé au chapitre 5.

Par rapport aux places de marché existantes que nous avons décrites dans la section 1.2, Kinvo se positionne ainsi :

- les clients sont des entreprises. Il s'agit de PME pour les vendeurs et d'investisseurs pour les

clients. On est donc dans un cadre B2B,

- le protocole est décrit dans le Chapitre 4. Nous en donnons plusieurs implémentations possibles. Dans tous les cas, il s'agit d'une variante du *bargaining* ayant un nombre fixe de tours. L'ensemble des variantes de ce protocole le classent dans les catégories 1:1 et 1:n. Le nombre de tours considéré est fixe,
- nous considérons le cas multi-attributs, ainsi que le cas mono-attribut. Le protocole que nous avons conçu supporte ces deux cas, et l'agent de négociation présenté dans la Chapitre 5 est capable de négocier dans ces deux contextes. Néanmoins, notre protocole impose une limitation dans le cas multi-attribut : il ne peut s'agir que d'attributs quantitatifs, et il est préférable qu'un prix en fasse partie,
- les agents sont dotés de fonctions d'utilité, *i.e.* de préférences souples.

En ce qui concerne les aspects techniques, notre plate-forme est implémentée, comme beaucoup d'autres, en Java. En revanche, les problématiques métier imposent une architecture matérielle et réseau particulière. D'autre part, l'architecture est couplée à Kyriba App, le portail de gestion de trésorerie de Kyriba. L'intégration d'un système multi-agents n'est donc pas évidente, et c'est à cette fin que nous avons conçu une ontologie, qui éclaire les liens entre la plate-forme d'une part et le système multi-agents de négociation d'autre part. Cette ontologie sert de point de départ au développement de notre protocole aussi bien que de notre agent, en définissant ses besoins.

Chapitre 2

Protocoles de négociation

“
It tells you the truth. As for how to read it, you'll have to learn by yourself.
Phillip PULLMAN, *The Golden compass*, 1995
”

Sommaire

2.1 Introduction 36
2.2 Protocoles de négociation 37
2.2.1 Enchères 37
2.2.2 *Bargaining* 42
2.2.3 Réseaux contractuels 44
2.2.4 Modèle producteurs-consommateurs 46
2.2.5 Discussion 52
2.3 *Mechanism design* 53
2.3.1 Principes 53
2.3.2 *Mechanism design* et *bargaining* 58
2.3.3 Discussion 64
2.4 Conclusion 65

2.1 Introduction

Supposons que vous ayez un bien à vendre. Comment vous y prendriez-vous ? Cette question peut sembler absurde, et pourtant, dans notre vie quotidienne, il existe de nombreuses manières d'acheter et de vendre, dont certaines proposent à l'acheteur de discuter directement avec le vendeur, encourageant les deux parties à négocier de manière bilatérale, tandis que d'autres implémentent un système d'enchère, géré par une place de marché.

Il est impossible de négocier sans suivre un certain nombre de règles. L'ensemble des règles suivies par les agents qui négocient est appelé « protocole ». Le protocole définit ce que l'agent a « le droit » de faire. Ainsi, il est impossible au cours d'une enchère anglaise (classique) de faire une proposition dont le prix est inférieure à la dernière proposition qui a été faite. Cette action ne fait pas partie des actions légalement réalisables par un agent dans ce type d'enchères.

Lorsqu'on parle de négociation, les protocoles les plus régulièrement utilisés comportent les différents types d'enchères, le *bargaining* (marchandage), les réseaux contractuels et les modèles producteurs-consommateurs, aussi appelés mécanismes de double enchère. Ces protocoles ont différentes caractéristiques, et donc différents avantages et inconvénients. Ils sont généralement analysés de manière axiomatique. Chaque protocole a ainsi un certain nombre de propriétés. [Sandholm, 1999] et [Kraus, 2001] donnent la liste suivante :

bien-être social Le protocole doit être optimal au sens d'un bien-être social, c'est à dire maximiser une certaine agrégation (somme, produit...) d'utilité ou des profits des agents,

efficacité de Pareto Le protocole mène à une situation où améliorer l'utilité ou le profit d'un agent se fait forcément aux dépens d'un autre agent,

rationalité individuelle Un agent a plus intérêt à négocier avec ce protocole plutôt que ne pas négocier du tout,

incitation Un agent a plus intérêt à faire une action particulière plutôt qu'aucune autre,

efficacité calculatoire Le protocole demande peu de calcul de la part des agents,

efficacité de communication Le protocole demande peu d'échanges entre les agents,

distribution Le protocole n'implique pas de rôle central.

Certains de ces critères peuvent cependant s'opposer, ou tout du moins entrer en concurrence. Une bonne illustration de ces dilemmes est le domaine des enchères combinatoires, dont nous traitons dans la Section 2.2.1.2. Le calcul d'une solution maximisant le bien-être social y est complexe, et va à l'encontre du critère d'efficacité de calcul. Dans un tel contexte, il est nécessaire soit de restreindre les utilités possibles des agents, soit d'accepter de trouver une solution qui atteindra un bien-être social non optimal, soit d'accepter de faire un calcul qui explosera en une complexité exponentielle. Il est impossible, dans le cas général, d'obtenir à la fois une complexité polynômiale et une allocation des biens qui soit optimale.

Il est possible d'analyser les protocoles rigoureusement en y implémentant des mécanismes. L'analyse porte en général sur des axiomes tels que ceux cités ci-dessus. Les propriétés de ces protocoles peuvent alors être définies de manière mathématique. Il est ensuite possible d'une part d'analyser les protocoles existants et de vérifier s'ils répondent à ces critères et d'autre part de réutiliser ces protocoles en les modifiant afin qu'ils valident certains axiomes par construction. Ce sont ces deux objectifs que remplit le *mechanism design*, une sous-discipline de l'économie.

Avant de nous intéresser plus avant à la manière d'analyser les protocoles de négociation, nous commençons par donner en exemple les protocoles les plus communs. Une fois ces protocoles définis,

nous proposons de présenter le *mechanism design*. Dans notre contexte, nous nous intéressons plus particulièrement au protocole concernant notre place de marché, à savoir le *bargaining*.

2.2 Protocoles de négociation

Négocier est un acte paradoxal, à la fois coopératif et compétitif. Lorsque deux agents négocient, ils cherchent à trouver un accord qui leur soit mutuellement profitable. Dans le même temps, la négociation ne peut être considérée comme purement collaborative. Les agents négocient parce qu'il existe un point de désaccord entre eux. Cette situation complexe impose de procéder en suivant un certain nombre de règles, qui constituent des protocoles.

Les plus anciennes traces de négociations relevées remontent à la préhistoire [Ndiema *et al.*, 2010], et certains protocoles existent donc depuis cette époque. Cependant, le développement des mathématiques économiques, dont certains résultats sont présentés dans la Section 2.3, a donné lieu à une analyse plus systématique. Nous définissons dans cette section un ensemble de protocoles classiques, couramment utilisés dans les systèmes multi-agents. Ces protocoles ont chacun des caractéristiques et des propriétés que nous démontrons, et nous présentons également un certain nombre de variantes pour chacun d'entre eux.

Du fait de ces propriétés, on associera à chaque problème de négociation particulier un ou plusieurs protocoles qui répondront au mieux au problème.

2.2.1 Enchères

Lorsqu'on parle d'enchères, nombre de personnes ne pensent qu'aux enchères anglaises, où chacun fait une proposition connue de tous et plus importante que la précédente. Pourtant, il existe de nombreux types d'enchères.

Une enchère est un protocole qui permet d'acheter un bien en faisant dessus des offres, appelées également enchères, dessus. L'agent ayant fait la meilleure offre – la plus haute – remporte alors le bien. Le processus est en général contrôlé par un tiers appelé « commissaire priseur » ou plus simplement « commissaire ».

On distingue deux types d'enchères : les enchères dites « simples » se font sur un unique bien à la fois. Les enchères dites « combinatoires » sont une généralisation où les agents font des offres sur des ensembles de biens qui peuvent être, par exemple complémentaires ou au contraire substituables. Nous laissons de côté les enchères dites « doubles » qui constituent un domaine à part, traité dans la Section 2.2.4.

2.2.1.1 Enchères simples

Les enchères simples se font sur un bien à la fois. Chaque agent acheteur fait une ou plusieurs offres sur le bien et celui qui fait la meilleure offre achète finalement le bien. Malgré ce principe commun, il existe plusieurs types d'enchères. Les plus répandues sont les suivantes :

L'enchère ascendante ou enchère anglaise est une enchère itérative au cours de laquelle chaque agent peut faire une proposition, tant que celle-ci est plus haute que la dernière qui a été faite.

Lorsque aucun agent ne souhaite plus faire de proposition, la dernière remporte l'enchère et celui qui l'a faite paie le dernier prix qu'il a annoncé.

L'enchère descendante ou enchère hollandaise est également une enchère itérative au cours de laquelle le commissaire annonce une série descendante de prix jusqu'à ce qu'un agent arrête la série. Cet agent remporte alors l'enchère et paie le dernier prix annoncé.

L'enchère scellée au premier prix est une enchère à offre unique au cours de laquelle chaque agent fait de façon simultanée une offre que les autres agents ne peuvent pas voir. L'agent faisant l'offre la plus haute remporte l'enchère et paie le prix qu'il a annoncé.

L'enchère de Vickrey fonctionne de la même façon que la précédente, mais le gagnant paie le second meilleur prix. Comme le montre le Théorème 2.1, cette particularité a pour avantage de pousser le joueur à donner sa véritable évaluation du bien, ni plus ni moins.

Une des problématiques majeures des enchères est d'inciter les agents à donner leur véritable évaluation d'un bien, c'est à dire d'être honnête [1].

Les protocoles d'enchères ne sont généralement pas incitatifs. Certains protocoles décrits ci-dessus ne donnent aucune incitation, tandis que d'autres incitent les agents à faire une offre au-dessous de leur estimation. Seule l'enchère de Vickrey incite à l'honnêteté. Cette propriété a été démontrée par [Vickrey, 1961] et est reproduite dans le Théorème 2.1. La stratégie optimale dans l'enchère anglaise est de faire une proposition toujours un peu plus haute que la précédente ; ainsi, l'offre la plus haute correspond à la seconde plus haute évaluation du bien.

Dans les enchères hollandaise et scellée au premier prix, il n'y a pas de stratégie optimale (pas d'incitation) mais il est possible pour un agent de tenter d'obtenir le bien à un prix plus faible que celui auquel il l'évalue.

Théorème 2.1 (Incitation à l'honnêteté dans l'enchère de Vickrey [Vickrey, 1961])
Dans le cadre d'une enchère de Vickrey, un agent a intérêt à donner sa véritable évaluation du bien qui lui est proposé.

Démonstration. Supposons qu'un agent donne un prix plus faible de ε que son évaluation du prix π :

- S'il remporte l'enchère, alors, le second prix le plus important π' vérifie : $\pi' < \pi - \varepsilon < \pi$. L'agent paie π' ; dans le cas où il fait une enchère à π, il remporte également l'enchère et paie autant.
- S'il ne remporte pas l'enchère, notons π' l'offre qui a remporté l'enchère ; alors :
  - Si $\pi' > \pi$, si l'agent fait une enchère au prix π, il perd aussi l'enchère et arrive au même résultat.
  - Si $\pi > \pi' > \pi - \varepsilon$, alors, si l'agent fait une offre au prix π, l'agent obtient le bien à un prix de $\pi' < \pi$, ce qui est plus avantageux pour lui, puisqu'il obtient le bien à un prix plus petit que son prix de réserve.

Donc, donner un prix plus faible que celui auquel on a évalué le bien n'est pas intéressant.

Supposons maintenant qu'un agent donne un prix plus haut de ε que son évaluation du prix π :

- S'il remporte l'enchère, alors :
  - Si le second prix le plus important π' vérifie : $\pi' < \pi < \pi + \varepsilon$, l'agent paie π' ; dans le cas où il fait une offre au prix π, il remporte aussi l'enchère et paie autant.
  - Si le second prix le plus important π' vérifie : $\pi < \pi' < \pi + \varepsilon$ l'agent paie π', c'est à dire plus cher que son prix de réserve. S'il fait une offre au prix π, il perd l'enchère, ce qui est meilleur pour lui que de payer plus que son prix de réserve.

1. le lecteur pourra se référer à [Klemperer, 1999] pour plus d'information sur les propriétés mathématiques des enchères

- S'il ne remporte pas l'enchère, notons π' l'offre qui a remporté l'enchère ; alors $\pi' > \pi + \varepsilon > \pi$, donc l'agent perd l'enchère qu'il fasse une offre au prix π ou au prix $\pi + \varepsilon$

Donc, donner un prix plus haut que celui auquel on a évalué le bien n'est pas intéressant.

Donc, la stratégie optimale d'un agent dans le cadre d'un protocole de Vickrey est de faire une offre au prix de son évaluation de ce bien.

□

Ces problématiques théoriques ne sont pas les seules qui se posent dans les mécanismes d'enchères. Un autre problème peut apparaître si les acheteurs communiquent entre eux. Ils peuvent alors s'entendre afin de faire baisser le prix d'un objet en vente. Certains des protocoles précédemment définis favorisent de telles collusions, tandis que d'autres les découragent. Quand des acheteurs s'entendent, l'acheteur qui a la plus haute évaluation pour un bien (et que nous appellerons « premier acheteur ») l'obtiendra, mais les autres acheteurs peuvent baisser leur prix de façon que le premier acheteur obtienne ce bien à un prix moindre que celui qu'il aurait payé si les acheteurs ne s'étaient pas entendus. Cette stratégie est tenable si dans le protocole, le fait de trahir les autres acheteurs n'est pas intéressant pour un agent. L'enchère de Vickrey et l'enchère anglaise ne sont pas tolérantes à de telles manipulations. Pour les deux autres types d'enchères décrits ci-dessus, une coalition n'est stable que si le premier acheteur fait une offre entre son prix et le second meilleur prix. Dans le cas contraire, la coalition est instable au sens où la trahison d'un des membre – et donc le fait d'enchérir entre le prix réduit et son prix de réserve – peut amener le traître à emporter l'enchère à un prix inférieur à son prix de réserve. Nous reprenons les démonstrations de [Robinson, 1985] dans les Théorèmes 2.2 à 2.4. Pour tous ces théorèmes, on suppose les acheteurs rationnels, *i.e.* l'agent i préfère payer π et gagner l'enchère à ne rien obtenir si et seulement si π est plus bas que son prix de réserve.

Théorème 2.2 (Stabilité d'une coalition d'acheteurs dans l'enchère anglaise)
Si les acheteurs sont rationnels, dans l'enchère anglaise, la coalition consistant à faire une offre au prix π_1 pour le premier acheteur et $\pi < \pi_1$ pour les autres acheteurs est stable.

Démonstration. Supposons qu'un acheteur i trahisse la coalition et fasse une offre $\pi' > \pi_1$. Notons rp_1 et rp_i les prix de réserve respectifs du premier acheteur et de l'acheteur i. On a alors $rp_1 > rp_i$. De plus l'acheteur est rationnel, donc $rp_i > \pi'$ donc $rp_1 > rp_i > \pi'$. En enchères anglaise, le premier acheteur peut faire une proposition au prix π'_1 vérifiant $rp_1 > \pi'_1 > \pi'$ et donc, l'agent i n'obtient pas le bien. Il n'a donc pas intérêt à faire la proposition π'. □

Théorème 2.3 (Stabilité d'une coalition d'acheteurs dans l'enchère de Vickrey)
Si les acheteurs sont rationnels, dans l'enchère de Vickrey, la coalition consistant pour le premier acheteur à faire une offre à son prix de réserve prix rp_1 et pour les autres à faire une offre $\pi < rp_1$ pour les autres acheteurs est stable.

Démonstration. Supposons qu'un acheteur i trahisse la coalition et fasse une offre $\pi' > \pi$. Notons rp_i les prix de réserve respectifs de l'acheteur i. On a alors $rp_1 > rp_i$. De plus l'acheteur est rationnel, donc $rp_i > \pi'$ donc $rp_1 > rp_i > \pi'$ et le premier acheteur gagne l'enchère malgré tout. L'agent i n'obtient pas le bien, il n'a donc pas intérêt à faire la proposition π'. □

Théorème 2.4 (Stabilité d'une coalition d'acheteurs dans l'enchère hollandaise)
Si les acheteurs sont rationnels, dans l'enchère hollandaise ou scellée au premier prix, la coalition consistant pour le premier acheteur à faire une offre à un prix π_1 et pour les autres à faire une offre

$\pi < \pi_1$ pour les autres acheteurs est stable si et seulement si $\pi_1 \geq rp_2$ le second prix de réserve le plus important.

Démonstration. Nous démontrons que si $\pi_1 \geq rp_2$, alors la coalition est stable, et si $\pi_1 < rp_2$, la coalition n'est pas stable.

Supposons que $\pi_1 \geq rp_2$. Alors, il n'y a aucun π' vérifiant $rp_i > \pi' > \pi_1$ où rp_i est le prix de réserve de l'agent i. L'agent i ne peut donc pas faire d'offre rationnelle qui change l'issue de l'enchère.

Supposons que $\pi_1 < rp_2$ et qu'un acheteur i trahisse la coalition et fasse une offre $rp_i > \pi' > \pi$. Alors, si l'agent i fait une offre au prix π', il obtient le bien au prix $\pi' < rp_i$, ce qui est une meilleure situation que ne pas obtenir le bien du tout. □

Les enchères simples sont aujourd'hui utilisées dans nombre de situation. Dans un cadre compétitif, elles sont utilisées dans les sites de e-commerce, comme ebay [eba, 1995]. Du fait de leur simplicité et de leur propriété d'incitation, l'enchère de Vickrey et ses variantes sont utilisées dans pour l'achat d'emplacements publicitaires sur les sites Internet [Varian et Harris, 2013]. Les enchères peuvent aussi être implémentées de façon décentralisée, mais dans un cadre collaboratif, en décentralisant le rôle du commissaire priseur [Choi *et al.*, 2009]. Dans un cadre compétitif, un commissaire est nécessaire, et le protocole est centralisé.

En ce qui concerne l'efficacité de Pareto de ces protocoles, les enchères anglaise et de Vickrey sont optimales, comme montré par [Vickrey, 1961], dans le cas où les valeurs des biens mis aux enchères sont indépendantes. Cela n'est pas le cas des autres enchères décrites ci-dessus étant donné qu'il n'y a pas pour ces dernières de stratégie dominante.

2.2.1.2 Enchères combinatoires

Dans le cas où les préférences des agents sur ces biens ne sont pas additives, les protocoles n'attribuent pas nécessairement les biens à ceux qui le désirent le plus. La Table 2.1 expose un tel cas. Dans cet exemple, deux agents (1 et 2) enchérissent sur des biens a et b. Ils évaluent la valeur des biens indépendamment, mais aussi les ensembles de biens (dans ce cas, $\{a, b\}$), qui n'est pas nécessairement la somme des évaluations de a et de b. Dans le cas décrit, l'agent 1 obtient les deux biens pour un prix de 4 dans une enchère anglaise, et de 3 dans le cadre d'une enchère de Vickrey, alors que l'agent 2 évalue le prix de l'ensemble à 5. Ce type de problématiques apparaît lorsque les préférences des agents ne sont pas additives, *i.e.* lorsque l'évaluation d'un ensemble de biens n'est pas égal à la somme des évaluations de chaque bien individuellement.

agent	$\{a\}$	$\{b\}$	$\{a, b\}$
1	2	2	4
2	1.5	1.5	5

TABLE 2.1 – Évaluations d'agents sur tous les lots de biens possibles

Une façon de contourner les limitations des enchères simples est d'utiliser un autre type de protocoles qui prend en compte les utilités non additives. C'est ce que permettent les enchères combinatoires, décrites notamment par [de Vries et Vohra, 2003, Cramton *et al.*, 2006]. Elles permettent d'enchérir sur des ensembles (appelés « *bundles* ») de biens. Ce type d'enchères amène plusieurs problématiques, notamment la détermination d'une allocation des biens optimale pour le vendeur. Du point de vue du

mécanisme, une autre problématique porte sur le moyen de communiquer les enchères des acheteurs au vendeur.

Ce second problème est particulier aux enchères combinatoires. Comme son nom l'indique, cette famille de protocoles pose un problème du point de vue des enchères envoyées par les agents. En effet, pour un ensemble de n biens, chaque agent fait une offre pour chacun des *bundles* possibles, soit au total 2^n enchères. Lorsque de nombreux biens sont mis en vente, la taille du message envoyé peut devenir trop importante pour que l'envoi ne soit pérenne. Il existe cependant plusieurs alternatives possibles pour palier ce problème :

- Il est d'abord possible de permettre aux acheteurs d'envoyer un *oracle*, une boîte noire fournissant une proposition pour chaque *bundle* de biens.
- On peut également définir un format d'enchères efficace, comme proposé dans [Nisan, 2000].
- Enfin, il est possible de réduire la taille du problème par l'imposition de certaines restrictions sur les possibilités d'enchères sur les *bundles*.

Le premier problème, qui porte sur l'allocation optimale des biens est plus compliqué à traiter. Ainsi, en notant $\mathcal{M}$ l'ensemble de tous les biens et $\mathcal{S} = \mathcal{P}(\mathcal{M})$ [2], $b_x(S)$ la valeur de l'enchère d'un agent x sur le *bundle* S et $\mathcal{N}$ l'ensemble de tous les agents, Sandholm montre dans [Sandholm, 2002] que l'allocation optimale des biens en fonction des enchères peut être formulée ainsi :

$$\max \sum_{x \in \mathcal{N}} \sum_{S \in \mathcal{S}} b_x(S) \times a(S, j) \tag{2.1}$$

tel que

$$\begin{cases} \forall i \in \mathcal{M}, \sum\limits_{\{S \in \mathcal{S} | i \in S\}} \sum\limits_{x \in \mathcal{N}} a(S, x) \in \{0, 1\} \\ \forall x \in \mathcal{N}, \sum\limits_{S \in \mathcal{S}} a(S, x) \in \{0, 1\} \\ \forall (S, x) \in \mathcal{S} \times \mathcal{N}, a(S, x) \in \{0, 1\} \end{cases}$$

Cette définition montre que l'allocation optimale des biens est identifiable au problème du sac à dos, un des 21 problèmes de Karp, donnés dans [Karp, 1972], et qu'il est donc NP-Complet. La littérature autour du problème du sac-à-dos est dense, et beaucoup de méthodes permettent de le résoudre approximativement (voir par exemple [Hazan *et al.*, 2006]) ou d'en résoudre des cas spéciaux, comme montré par [Parker et Rardin, 2014]. Sa résolution exacte dans le cas général est cependant combinatoire (à moins que P=NP).

Le fait d'enchérir sur un ensemble de biens plutôt que sur chaque bien séparément ne suffit pas à définir un protocole d'enchères. Il est toujours nécessaire de choisir comment les agents enchérissent. De la même façon que les enchères simples, plusieurs mécanismes sont régulièrement utilisés :

Enchères à une étape Dans ce type d'enchères, les agents envoient une proposition pour chaque *bundle* possible (voire alternativement un oracle permettant de fournir de telles propositions au commissaire priseur).

Modification itérative de la quantité Dans ce type d'enchères, les agents proposent des *bundles* et un prix pour ces *bundles* (propositions utilisant des *OU* exclusifs) et envoient ces propositions au commissaire. Le commissaire fait une allocation provisoire, en s'appuyant sur les propositions ; les agents acheteurs peuvent par la suite modifier leurs propositions. Le mécanisme doit être conçu de façon à encourager la compétition et accélérer la négociation.

Modification itérative du prix Dans ce type d'enchères, les acheteurs font des propositions sur les *bundles*, et le commissaire en ajuste le prix prenant en considération toutes les offres qui ont

2. On note $\mathcal{P}(A)$ l'ensemble des sous-ensembles de A.

été faites. Ce type d'enchères a pour objectif d'allouer un maximum de biens, et d'équilibrer offre et demande.

Les deux derniers types de protocoles sont appelés enchères combinatoires itératives, puisqu'ils prennent plusieurs étapes pour arriver à une solution. L'avantage et d'éviter de considérer tous les *bundles* possibles, et de ne prendre en compte que ceux auxquels les agents sont vraiment intéressés. Le lecteur intéressé pourra se référer au second chapitre de [Cramton *et al.*, 2006] ainsi qu'à [Parkes et Ungar, 2000] pour plus d'informations sur les enchères combinatoires itératives.

Les enchères combinatoires sont généralement utilisées dans les cas où il y a une forte dépendance entre biens. C'est par exemple le cas lorsque des pistes d'un aéroport sont attribuées à des compagnies aériennes [Rassenti *et al.*, 1982]. Comme les enchères simples, les enchères combinatoires peuvent aussi être utilisées de manière décentralisée dans les cas collaboratifs, comme le montrent [Choi *et al.*, 2009]

Les protocoles d'enchères sont des protocoles impliquant un tiers de confiance. Il laissent assez peu de liberté aux vendeurs. Ils sont généralement utilisés dans le cas où les biens sont intervertibles, comme dans le cas de la vente de poisson ou quand le vendeur est neutre vis-à-vis des vendeurs, comme lors de la vente des « fréquences en or » en 2015. L'un des problèmes majeurs des enchères est que le vendeur a finalement un rôle très limité. Une fois son bien en vente, il ne peut plus intervenir ni interagir avec aucun des agents. Il n'a notamment aucun contrôle sur l'acheteur qui achète son bien. Dans le cas de notre place de marché, nous souhaitons qu'une relation de confiance puisse s'établir entre le créancier et le fournisseur. Les enchères sous toutes leurs formes ne semblent donc pas être les protocoles les plus à même de répondre à notre problématique

2.2.2 *Bargaining*

Lorsque le vendeur veut conserver un contrôle sur l'acheteur auquel il vend son bien, il utilise une autre famille de protocoles : les protocoles de *bargaining*, notamment décrits dans la Section 5 de [Sandholm, 1999]. Le *bargaining* est un protocole impliquant deux agents cherchant à trouver un accord dans un espace d'accords possibles. À cette fin, les deux agents échangent des propositions. Ces dernières sont des éléments de l'ensemble des accords possibles. À chaque fois qu'un agent reçoit une proposition, il peut choisir soit de l'accepter, soit de la refuser, soit de faire une contre-proposition. Dans le premier cas, le résultat de la négociation est la proposition qui a été acceptée. Dans le second cas, la négociation s'arrête et rien ne se passe. Enfin, une contre-proposition est une proposition de l'agent, qu'il renvoie à son partenaire de négociation.

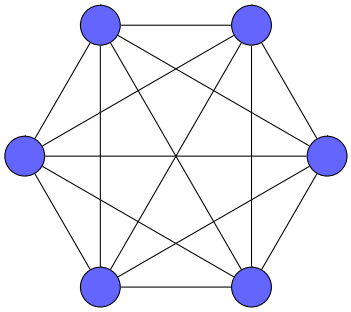
L'un des aspects capitaux du *bargaining* est qu'il ne demande pas aux agents de révéler leurs informations sensibles telles que leur utilité à leur partenaire. Cela crée une difficulté à trouver un accord, en particulier dans le cas où le domaine de négociation est large. Les agents ne savent pas quel est le domaine des propositions acceptables pour l'adversaire, ni même quelles sont ses préférences. Cela peut être inféré en utilisant l'*opponent modeling*, introduite dans le Section 3.2 mais cette modélisation n'est jamais parfaite.

Lorsque les agents cherchent à cacher leurs informations privées de leur partenaires, mais qu'ils sont prêts à les confier à quelqu'un d'autre, il est possible d'introduire un tiers de confiance. Dans la Section 2.3.2 nous donnons l'exemple d'un tel protocole qui utilise un tiers de confiance qui reçoit les utilités des agents et détermine une solution de manière unilatérale.

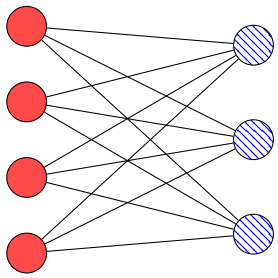
L'aspect itératif du *bargaining* disparaît alors complètement. Cependant, ajouter un tiers de confiance ne signifie pas nécessairement abandonner l'aspect itératif du *bargaining*. [Wilson, 2001]

propose un protocole de *bargaining* avec un médiateur. Ce dernier reçoit les informations privées – typiquement les fonctions d'utilité – des agents et génère à partir de ces informations une proposition qu'il leur soumet. Ces derniers peuvent l'accepter ou la refuser, auquel cas le médiateur en génère une nouvelle. Le but est de trouver un accord plus aisément tout en laissant le pouvoir de décision aux agents négociateurs.

Les procédures de *bargaining* peuvent se différencier selon que l'objet négocié soit perçu comme étant mono-attribut ou multi-attributs. En *bargaining* classique, les agents ne s'intéressent qu'au prix de vente du bien. [Blume et Durlauf, 2008, Winoto, 2007], par exemple, dotent chaque agent d'un prix de réserve : l'utilité de l'agent est alors la différence entre son prix de réserve et le prix négocié. Par ailleurs, plusieurs travaux *e.g.* [Fershtman, 1990, Fershtman, 2000, Inderst, 2000] ont introduit le *bargaining* multi-attributs (ou multi-problématiques) ; ces attributs sont en général le prix et la qualité, mais peuvent dépasser ce simple cadre. Nous discutons des agents négociant dans ce contexte dans le Chapitre 3. L'utilité de l'agent est alors exprimée comme une agrégation des attributs.



(a) Protocole de bargaining multilatéral de [Osborne et Rubinstein, 1990]



(b) Protocole de bargaining multilatéral de [An *et al.*, 2016]

FIGURE 2.1 – Deux variantes de *bargaining* multilatéral

Une autre variation porte sur la négociation multilatérale. Comme le montre la Figure 2.1, il y a plusieurs moyens de rendre le *bargaining* multilatéral. Le premier consiste à impliquer plus d'agents dans la négociation d'un bien. Les rôles d'acheteur et de vendeur que nous utilisons généralement disparaissent alors afin de laisser place à un contexte plus général où plusieurs agents font des propositions tour à tour afin de trouver un accord. C'est notamment ce que proposent [Osborne et Rubinstein, 1990]. Ce protocole a été repris dans la sixième édition de l'*Automated Negotiation Agent Competition* en 2015. Il a également été utilisé pour répartir des ressources naturelles (*cf.* [Simon *et al.*, 2003]). Cette problématique est orthogonale aux caractéristiques des biens échangés et au fait qu'ils peuvent être mono ou multi-attributs.

Mais il y a une autre façon de rendre le *bargaining* multilatéral : faire plusieurs négociations en parallèle. C'est ce que proposent [An *et al.*, 2009, An *et al.*, 2016]. Leur protocole est identique à plusieurs *bargainings* qui auraient lieu de manière simultanée. Cependant, mener ces *bargainings* en parallèle implique que plusieurs acheteurs peuvent accepter une proposition à propos du même bien au même moment, et que plusieurs vendeurs peuvent accepter une proposition du même acheteur pour un bien similaire. Le protocole se voit donc adjoindre une étape supplémentaire après l'`accept`, permettant à l'agent d'annuler la transaction s'il le souhaite. La Figure 2.2 montre le diagramme séquence AUML (Agent UML, voir [Bauer *et al.*, 2001]) du protocole.

Les protocoles de *bargaining* sont généralement avantageux lorsque les agents souhaitent négocier directement les uns avec les autres, et permettent ainsi aux vendeurs et aux acheteurs d'opérer un

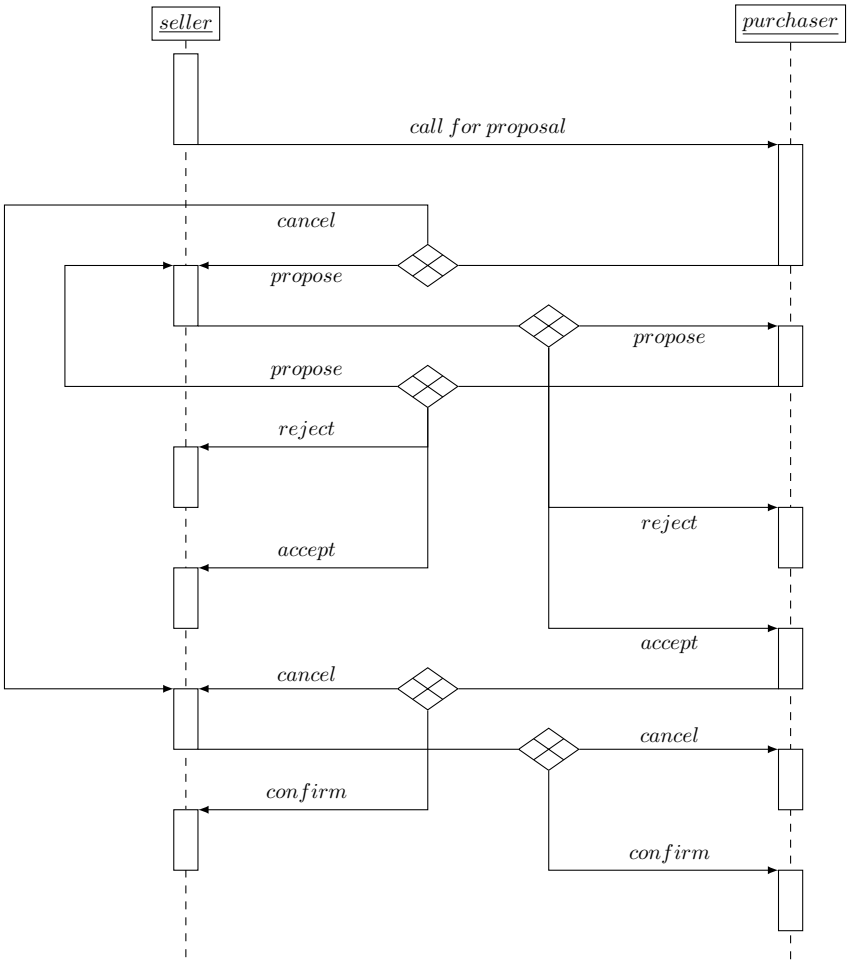


FIGURE 2.2 – Diagramme de séquences du *bargaining* multitiers

choix, et ainsi de développer des partenariats. Il n'en reste pas moins qu'à l'exception des protocoles avec médiateurs décrits dans la Section 2.3.2, ils impliquent un échange d'information conséquent sur des informations qui sont censées rester secrètes. Si ces informations doivent rester secrètes, c'est qu'elles sont privées et que leur connaissance peut faire du tort à l'agent sur lequel elles portent. Cette problématique est traitée dans le Tableau 2.4.

2.2.3 Réseaux contractuels

Les réseaux contractuels ont été définis pour la première fois par [Davis et Smith, 1983, Smith, 1980]. Le principe est la délégation des tâches : un gestionnaire (*manager*) a une tâche à accomplir, et la divise en plusieurs sous-tâches. Il envoie un appel à proposition à tous les autres agents (appelés contractants *contractors*). Les contractors qui peuvent traiter ces sous-tâches répondent et envoient une offre au manager. Les autres envoient un refus. Le manager choisit ensuite parmi les offres celle qui est la plus intéressante de son point de vue. Il envoie un message `accept` à son expéditeur et un `reject` aux autres. Le contractor informe enfin le manager de son succès – et si besoin en est du

résultat – ou de son échec. La Figure 2.3 donne une vision globale de ce protocole.

[Smith, 1980] décrit techniquement le réseau contractuel. Son article décrit également différentes extensions possibles de ce protocole. La première est liée à la transmission des messages. En effet, dans les protocoles de réseaux contractuels, les agents envoient de nombreux messages, en particulier durant l'appel à propositions.

Une des extensions du *Contract Net Protocol* est l'*Iterative Contract Net Protocol*. Ce protocole est identique au *Contract Net Protocol* jusqu'au moment où l'agent manager renvoie un *accept* ou un *reject* aux contractors. Ces messages sont remplacés par un échange de type *bargaining*. Le diagramme de séquence AUML de l'*Iterative Contract Net Protocol* est donné en Figure 2.4.

D'autres variantes du *Contract Net* ont été proposées, notamment celle de [Sandholm et Lesser, 2001]. Cette extension règle le même problème que [An *et al.*, 2016] : il permet aux agents de se désengager du protocole. Outre le prix du contrat, [Sandholm et Lesser, 2001] proposent par ailleurs de faire payer à l'agent un montant calculé en fonction du moment où il se désengage. Cela permet de faire payer davantage les agents qui ne respectent pas leur engagement.

Les protocoles de réseaux contractuels se trouvent entre les protocoles d'enchère et les protocoles de *bargaining*. Ce sont des protocoles multilatéraux, mais où il reste une certaine liberté au vendeur quant à l'offre qu'il accepte. Ils n'échappent cependant pas au problème de l'intervention du vendeur

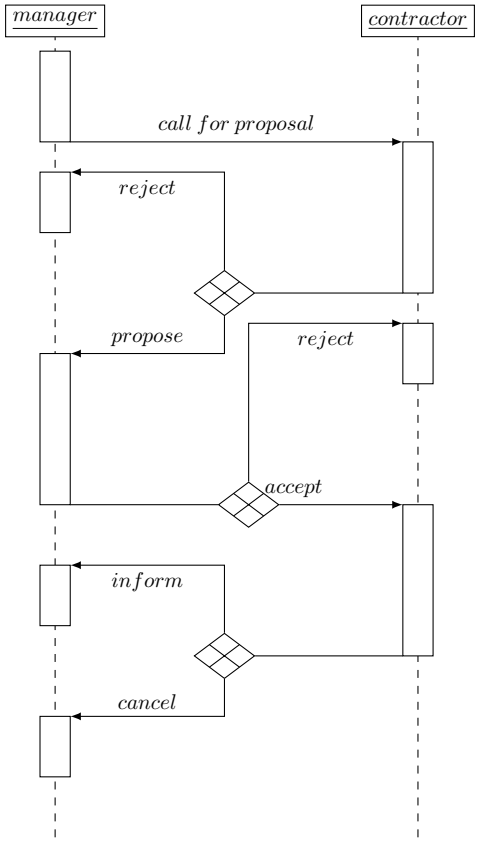


FIGURE 2.3 – Diagramme de séquences du protocole de réseau contractuel

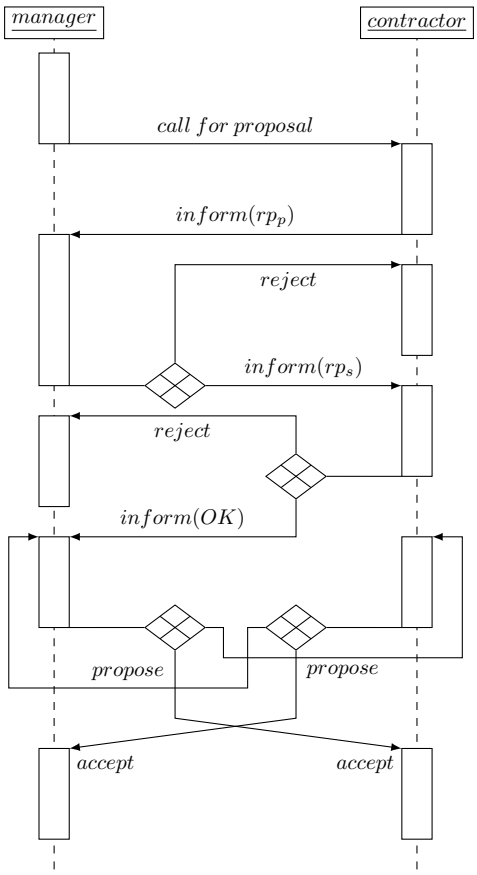


FIGURE 2.4 – Diagramme séquences de l'*Iterated Contract Net Protocol*

(ou du gestionnaire dans ce cas) : seul l'acheteur (le contracteur) est autorisé à faire des offres ; dans une situation où le domaine de négociation est large et où les fonctions d'utilité sont difficiles à inférer, l'acheteur risque de faire des concessions qui ne profiteront pas à l'adversaire. Dans le cadre de notre plate-forme, cela est problématique.

2.2.4 Modèle producteurs-consommateurs

Nous avons volontairement laissé de côté un type d'enchère dans la Section 2.2.1. Il s'agit des enchères dites « doubles ». Dans ce type d'enchères, un unique type de bien est échangé et les acheteurs comme les vendeurs font des propositions sur le prix de vente et d'achat. Le commissaire reçoit l'ensemble de ces prix et attribue ensuite au bien un prix. L'attribution de ce prix est axiomatique, et doit correspondre à un certain nombre de critères tels que définis dans la Section 2.3.1. C'est ce type d'équilibre que nous développons par la suite.

2.2.4.1 Définitions

On définit l'équilibre général dans un marché. On note :

- $(g_1, \ldots, g_n)$ les n types de biens présents sur le marché, y compris le travail,
- $\mathbf{p} = (p_1, \ldots p_n)$ le prix de ces biens sur le marché,
- un ensemble de m producteurs et de r consommateurs.
- Y la matrice de production. Un élément y_i^k de cette matrice correspond à la quantité du bien $i \in [\![1, n]\!]$ que produit un producteur $k \in [\![1, m]\!]$ [3],
- $X = (X_1, \ldots, X_r)$ l'ensemble des *bundles* par lesquels le consommateur $i \in [\![1, r]\!]$ est intéressé. On a donc : $\forall i \in [\![1, r]\!], X_i = \{\mathbf{x}_1^i, \ldots, \mathbf{x}_{q_r}^i\}$ où $\forall i \in [\![1, n]\!], \forall j \in [\![1, q_i]\!], \mathbf{x}_j^i \in \mathbb{R}^n$. Il s'agit d'un vecteur de quantités de chaque bien qui constitue un *bundle* intéressant pour le consommateur.
- $U = (u_1 \ldots, u_r)$ l'ensemble des fonctions d'utilités des consommateurs. Pour un consommateaur $i \in [\![1, r]\!]$, u_i est définie sur l'ensemble des *bundles* qui l'intéressent. $u_i : X_i \to \mathbb{R}$. Cette fonction représente ce que l'agent est prêt à payer pour obtenir un *bundle*.
- $D = (D_1, \ldots D_r)$ la demande des consommateurs. Il s'agit de ce que demande concrètement le consommateur. Si l'acheteur est rationnel, cela dépend du vecteur de prix des biens. $\forall i \in [\![1, r]\!], D_j : \mathbb{R} \to X_j$ est une fonction qui à chaque vecteur prix associe le *bundle* que le consommateur r souhaite acheter. Pour $i \in [\![1, n]\!], j \in [\![1, r]\!]$ on note $D_j^i(\mathbf{p})$ la demande de j pour le bien i étant donné le vecteur de prix $\mathbf{p}$.
- $B = (\beta_{i,k})_{(i,k) \in [\![1,r]\!] \times [\![1,m]\!]}$ les coefficients de redistribution de richesse : on suppose que chaque producteur $k \in [\![1, m]\!]$ reverse au consommateur $i \in [\![1, r]\!]$ une part $\beta_{i,k}$ de ce qu'il a gagné, sachant que $\forall k \in [\![1, m]\!], \sum_{i \in [\![1,r]\!]} b_{i,k} = 1$, c'est à dire que l'intégralité de l'argent passe des producteurs aux consommateurs. On peut considérer que les producteurs sont des entreprises et les consommateurs des individus. Cette condition se traduit comme le fait que la même somme d'argent circule toujours.
- $E = (\mathbf{e}_1, \ldots, \mathbf{e}_r)$ les dotations initiales des consommateurs. $\forall i \in [\![1, r]\!], \mathbf{e}_i \in \mathbb{R}^n$ correspond au nombre d'unités que possède le consommateur i de chaque bien.

Définition 2.1 (Équilibre walrasien)
Un équilibre walrasien *[Walras, 1954] est une situation* $(\mathbf{p^{eq}}, (\mathbf{y_k^{eq}})_{k \leq m}, (\tilde{\mathbf{x}}_\mathbf{j}^\mathbf{eq})_{j \leq r})$ *dans laquelle :*

- *ce qui est produit est consommé, i.e.* $\sum_{j \leq r} \tilde{\mathbf{x}}_\mathbf{j}^\mathbf{eq} = \sum_{j \leq r} \mathbf{e_j} + \sum_{k \leq m} \mathbf{y_l^{eq}}$
- *les consommateurs sont rationnels, et cherchent à maximiser leur utilité i.e.* $\forall j \leq r, \tilde{\mathbf{x}}_\mathbf{j}^\mathbf{eq} = \underset{\mathcal{A}}{\text{argmax}}\ u(\mathbf{x_j})$ *où* $\mathcal{A} =$

$$\{\mathbf{x_j} \in X_j | \mathbf{p^{eq}} \cdot \mathbf{x_j} \leq \mathbf{p^{eq}} \cdot \mathbf{e_j} + \sum_{k \leq m} \beta_{j,k} \mathbf{p^{eq}} \cdot \mathbf{y_k^{eq}}\}$$

- *les producteurs sont rationnels et cherchent à maximiser leur revenu, i.e.* $\forall k \leq m, \mathbf{y_k^{eq}} = \underset{\mathbf{y_k} \in Y_k}{\text{argmax}}\ \mathbf{p^{eq}} \cdot \mathbf{y_k}$

Une littérature importante existe sur les théorèmes d'existence, d'unicité et de stabilité des équilibres généraux. Les travaux les plus marquants dans ce domaine sont ceux de [Walras, 1954, Arrow et Debreu, 1954, Debreu, 1974, Mantel, 1974, Sonnenschein, 1972]. Nous nous appuyons ici en particulier sur les travaux de [Arrow et Hurwicz, 1958], qui donnent les conditions d'existence et d'unicité de ces équilibres.

3. on a $y_i^k > 0$ si le bien est produit et $y_i^k < 0$ si le bien est consommé par la production de k

Théorème 2.5 (Existence d'un équilibre walrasien)
Supposons que :

- *la production de bien consomme d'autres biens (au moins du travail), i.e.*

$$\forall k \leq m, \begin{cases} \exists i \leq n : y_k^i < 0 \\ ou \\ \forall i \leq n, y_k^i = 0 \end{cases} \tag{2.2}$$

- $\forall k \leq m,\ Y_k$ *est fermé et convexe et* $\mathbf{0}_{\mathbb{R}^n} \in Y_k$
- *la production n'est pas réversible, i.e.* $\forall (k,k') \in [\![1,m]\!]^2, y_k \neq -y_{k'}$
- $\forall j \leq r,\ X_j$ *est fermé et convexe et* $\exists \mathbf{v_j} \in \mathbb{R}^n : \forall \mathbf{x_j} \in X_j, \mathbf{x_j} \geq \mathbf{v_j}$
- $\forall j \leq r,\ u_j \in \mathcal{C}^0(B_j, \mathbb{R})$
- $\forall j \leq r,\ u_j$ *est convexe, i.e.*

$$\begin{aligned} &\forall \gamma \in]0,1[, \forall (\mathbf{x_j}, \mathbf{x_j}') \in X_j^2, u_j(\mathbf{x_j}) > u_j(\mathbf{x_i}') \\ &\Rightarrow u_j(\gamma \mathbf{x_j} + (1-\gamma)\mathbf{x_i}') > u_j(\mathbf{x_i}') \end{aligned} \tag{2.3}$$

- $\forall j \leq r, \forall \mathbf{x_j} \in X_j, \exists \mathbf{x_j}' \in X_j : u_j(\mathbf{x_j}') > u_j(\mathbf{x_j})$

Alors, un équilibre walrasien existe.

Théorème 2.6 (Unicité de l'équilibre walrasien)
Soit une situation où les conditions précédentes sont vérifiées, et où on ajoute la condition de substituabilité de bien*, i.e. :*

$$\begin{aligned} &\forall (\mathbf{p}, \mathbf{q}), \textit{ vecteurs de prix tels que } \exists i \in [\![1,n]\!] : \begin{cases} p_i > q_i \\ \forall l \neq i, p_l = q_l \end{cases} \\ &\implies \forall l \neq i \sum_{j \in [\![1,r]\!]} D_l^i(\mathbf{p}) > \sum_{j \in [\![1,r]\!]} D_l^i(\mathbf{q}) \end{aligned} \tag{2.4}$$

La substituabilité des biens signifie que l'augmentation du prix d'un bien se traduit par une augmentation de la demande de tous les autres biens.

Sous ces conditions, il y a un et un seul équilibre walrasien.

L'efficacité (au sens de Pareto) de ces équilibres avait déjà été montrée par [Arrow, 1951].

Théorème 2.7 (Efficacité de Pareto)
Les équilibres généraux walrasiens sont Pareto-efficaces.

2.2.4.2 Recherche de l'équilibre de Walras

La première méthode qui a été proposée pour trouver l'équilibre général walrasien d'une économie est l'algorithme de tâtonnement. [Walras, 1954] propose une version de cet algorithme, mais une autre version, désormais considérée comme standard, avait déjà été proposée par [Samuelson, 1947]. Dans cette méthode, un *régulateur* centralise les plans de production et les demandes de consommation et ajuste les prix des biens de manière incrémentale. Cet algorithme est donné dans l'Algorithme 1. Le Théorème 2.8 stipule que cet algorithme amène a un équilibre walrasien *stable* (par stable, on entend que les prix convergent vers cet équilibre).

Algorithme 1 L'algorithme de tâtonnement

procedure RÉGULATEUR
  $\forall i \in [\![1, n]\!], p_i \leftarrow 1$ ▷ Initialisation des prix
  $\forall i \in [\![1, n-1]\!], \alpha_i \leftarrow Random()$
  while $\exists i \in [\![1, n]\!] : |\sum_{j \leq r} x_j^i - e_j^i - \sum_{k \leq m} y_k^i| > \varepsilon$ **do**
    $Broadcast(\mathbf{p}, \{consumers, producers\})$
    $\forall k \in [\![1, m]\!], production[k] \leftarrow receive(producer\ k)$
    $Broadcast(production, \{consumers\})$
    $\forall j \in [\![1, r]\!], consumption[j] \leftarrow receive(consumer\ j)$
    for $i \in [\![1, n-1]\!]$ **do**
      $p_i \leftarrow p_i + \alpha_i \cdot (\sum_{j \leq r} x_j^i - e_j^i - \sum_{k \leq m} y_k^i)$
    end for
  end while
  $Broadcast(\text{“}equilibrium'', \{consumers, producers\})$
end procedure

procedure PRODUCTEUR
  while $Receive(adjustor) \neq$ “$equilibrium''$ **do**
    $\mathbf{p} \leftarrow Receive(adjustor)$
    $production \leftarrow Receive(adjustor)$
    $send(consumption())$
  end while
  $Exchange()$
  $Consume()$
end procedure
function CONSOMMATION( ) :`double`$[n]$
  return $\arg\max_{\mathcal{A}} u(\mathbf{x_j})$ où $\mathcal{A} = \{\mathbf{x_j} \in X_j | \mathbf{p^{eq}} \cdot \mathbf{x_j} \leq \mathbf{p^{eq}} \cdot \mathbf{e_j} + \sum_{k \leq m} \beta_{j,k} \mathbf{p^{eq}} \cdot \mathbf{y_k^{eq}}\}$
end function

procedure CONSOMMATEUR
  while $Receive(adjustor) \neq$ “$equilibrium''$ **do**
    $price \leftarrow Receive(adjustor)$
    $Send(productionPlan(), auctioneer)$
  end while
  $Exchange()$
  $Produce()$
end procedure
function PLANDEPRODUCTION( ) :`double`$[n]$
  return $\arg\max_{\mathbf{y_k} \in Y_k} \mathbf{p^{eq}} \cdot \mathbf{y_k}$
end function

Théorème 2.8 (convergence du Tâtonnement)
Sous les conditions décrites précédemment, l'algorithme de tâtonnement comme décrit dans l'Algorithme 1 converge vers un équilibre walrasien.

Une autre problématique est l'implémentation concrète de cet algorithme. Une des propriétés les plus désirables de cette implémentation est la distribution. Il y a plusieurs niveaux de distribution [Cheng et Wellman, 1998] :

- au niveau de l'enchère, chaque bien a son propre régulateur,
- au niveau de l'agent, chaque agent est indépendant et ne connaît pas les préférences des autres agents,
- au niveau "géographique", certaines parties des marchés sont séparées des autres (ce niveau peut être utilisé si les coûts de communication sont élevés). Si les marchés locaux sont couplés, l'équilibre est le même que celui de l'algorithme centralisé. Si ce n'est pas le cas, les différents régulateurs auront différents prix pour le même bien.

Les algorithmes de tâtonnement ne sont pas les seules méthodes permettant de trouver les équilibres walrasiens. [Schlieper, 1974, Sambracos et Yannelis, 1993] donnent des résultats sur des processus de « non-tâtonnement ». Les processus de tâtonnement peuvent être assez longs, et dans certaines situations, attendre longtemps avant de commencer les échanges peut être problématique. Dans de telles situations, d'autres méthodes peuvent être utilisées. Ces méthodes, les processus de non-tâtonnement, amènent à des équilibres potentiellement non walrasiens, mais plus rapides que les processus de tâtonnement car ils permettent aux agents d'échanger avant que l'équilibre ne soit atteint.

Cette situation amène de nouvelles définitions :

- $\forall j \in [\![1, r]\!], \mathbf{s_j} = (s_j^i)_{i\in[\![1,n]\!]}$ est le stock de biens du consommateur j. On appelle aussi $\mathbf{s} = \sum_{j\in[\![1,r]\!]} \mathbf{s_j}$ le stock total des différents biens,
- pour toute variable v précédemment définie, on définit $\dot{v} = \dfrac{\partial v}{\partial t}$ la dérivée partielle de v par rapport au temps,

Définition 2.2 (Équilibre général)
Un équilibre général est une situation $(\mathbf{p}, \mathbf{s})$ *dans laquelle* $\forall j \in [\![1, r]\!], D_j(\mathbf{p}) = \mathbf{s_j}$.

Plusieurs approches ont été proposées : la première est appelée processus de Negishi [Negishi, 1961]. Dans cet article, Negishi décrit un marché dans lequel la substituabilité de bien est valide (*cf.* Théorème 2.6). Le marché respecte également le loi de Walras, *i.e.*

$$\sum_{i\in[\![1,n]\!]} p_i \cdot D_j^i(\mathbf{p}) - \sum_{i\in[\![1,n]\!]} p_i \cdot y_k^i = 0$$

Une autre présupposition sur le marché est que l'évolution des prix correspond à l'excès de demande de biens, *i.e.*

$$\dot{\mathbf{p}} = \sum_{j\in[\![1,r]\!]} D_j^i(\mathbf{p}) - \mathbf{s} \tag{2.5}$$

Finalement, on suppose que les transactions de crédit et les transferts unilatéraux sont interdits avant que l'équilibre ne soit atteint, *i.e.*

$$\sum_{i\in[\![1,n]\!]} p_i \cdot \dot{s}_i = 0$$

le Théorème 2.9 affirme que dans de tels marchés et sous certaines conditions, il est possible d'atteindre un équilibre stable même si les agents échangent avant que l'équilibre ne soit atteint.

Théorème 2.9 (Existence et unicité de l'équilibre dans le processus de Negishi [Negishi, 1961])
Sous les conditions suivantes :

- *tous les agents ont la même fonction d'utilité ($\exists u : \forall j \in [\![1, r]\!], u_j = u$),*
- *u est quasi-concave*
- *u est homogène*

Il y a un et un seul équilibre. De plus, cet équilibre est stable.

Le processus d'Edgeworth [Negishi, 1962] est proche du précédent, mais la condition de substituabilité des biens est retirée, et une autre est ajoutée sur la négociation : les seuls échanges autorisés en dehors de l'équilibre sont ceux qui sont Pareto-efficaces. Appelons $(\mathbf{s_j})_{j\in[\![1,r]\!]}$ l'état du stock au temps t, et $(\mathbf{s_j}')_{j\in[\![1,r]\!]}$ l'état du stock au temps $t + \Delta t$. Alors, $\forall j \in [\![1, r]\!], u_j(\mathbf{s_j}') \geq u_j(\mathbf{s_j})$.

Théorème 2.10 (Existence et unicité de l'équilibre du processus d'Edgeworth [Negishi, 1961])
Sous les mêmes conditions que le Théorème 2.9, le processus d'Edgeworth converge vers un équilibre stable.

Le processus de Hahn, introduit par [Hahn et Negishi, 1962] repose sur les mêmes principes mais les préconditions pour obtenir un équilibre stable sont différentes. Dans le processus de Hahn, la condition de Pareto-efficacité est retirée au profit d'une autre sur les échanges avant l'équilibre : s'il y a un excès de quantité d'un bien, on suppose qu'aucun agent ne recherche ce bien au prix actuel (sinon, il l'aurait acheté). Cette condition s'écrit :

$$\begin{aligned} \forall (i,j) \in [\![1,n]\!] \times [\![1,r]\!], (\tilde{x}^i - s^i) \cdot (\tilde{x}^i_j - s^i_j) \geq 0 \\ \forall (i,j) \in [\![1,n]\!] \times [\![1,r]\!], (\tilde{x}^i - s^i) = 0 \Leftrightarrow (\tilde{x}^i_j - s^i_j) = 0 \end{aligned} \tag{2.6}$$

L'utilité de chaque bien et les quantités de chaque bien sont supposées positives. De plus, $\forall j \in [\![1, r]\!] u_j$ doit être quasi-concave, *i.e.*

$$\begin{aligned} &\forall (\mathbf{x}, \mathbf{x}'), \forall \gamma \in [0,1], \\ &u_j(\mathbf{x}) > u_j(\mathbf{x}') \Rightarrow u_j(\gamma \mathbf{x} + (1 - \gamma \mathbf{x}') > u_j(\mathbf{x}') \end{aligned} \tag{2.7}$$

et $\forall j \in [\![1, r]\!], u_j \in \mathcal{C}^1$

Théorème 2.11 (Existence et unicité de l'équilibre du processus de Hahn [Hahn et Negishi, 1962])
Sous les conditions précédemment définies, le processus de Hahn converge vers un unique équilibre stable.

Fisher a amené deux nouveaux protocoles. Le premier, introduit par [Fisher, 1976, Fisher, 1983, Weintraub, 1979] utilise le processus de Hahn comme modèle de base (il est parfois appelé processus de Hahn-Fisher). Cependant, il apporte une nouveauté : il n'y a plus de régulateur. En fait, Fisher considère que chaque vendeur est capable de changer les prix des biens qu'il vend en fonction des demandes. Cette possibilité amène de nouvelles difficultés parmi lesquelles le risque de discontinuités et de seuils quand un acteur change sa fonction d'utilité. Ces effets sont d'autant moins importants qu'il y a plus d'agents.

Le second mécanisme, introduit par [Fisher, 1983, Tieben, 2009, Schinkel, 2001] amène la possibilité pour les agents de prendre en compte la prévision du futur. Les agents sont capables de percevoir qu'ils échangent dans une situation de déséquilibre et agir en conséquence. [Fisher, 1983] montre que sous ces conditions, il est possible d'arriver à un équilibre. Cependant, il est nécessaire de supposer deux conditions supplémentaires :

- les vendeurs pensent que les prix ne dépendent pas de leur propositions personnelles, et agissent en conséquence,
- les agents n'essaient pas de bénéficier de cette situation. Cette condition est appelée "pas de surprise avantageuse" dans [Fisher, 1983] et "pas de surprise statistique" dans [Schinkel, 2001]. Cette condition suppose que les agents n'utiliseront pas l'inflation ni n'achètent en vue de revendre plus tard à un prix supérieur, ce qui pourrait arriver puisqu'ils savent qu'ils sont en situation de déséquilibre (et donc que les prix changent).

Fisher souligne deux questions à propos de ses travaux : tout d'abord, comment un agent peut-il interpréter et prédire les changements de prix ? [Schinkel, 2001] donne une réponse possible à cette question. L'utilisation de l'apprentissage bayésien permet d'avoir une inférence sur l'évolution des prix. La seconde question consiste à comprendre l'équilibre vers lequel le mécanisme converge (et qui n'est pas walrasien), et la vitesse de convergence. Finalement, une des problématiques de ce mécanisme est qu'on suppose que le nombre d'entreprises est fixe et ne peut changer durant le processus.

Les modèles producteurs-consommateurs sont particulièrement utilisés pour modéliser et simuler l'économie d'un pays ou d'une institution, mais aussi comme protocole de coordination et de négociation. Le processus de Hahn a notamment été utilisé dans divers systèmes multi-agents. [Nakatsuka *et al.*, 2000] montrent par exemple comment utiliser ce processus dans les problèmes d'allocation de ressource.

Les modèles producteurs-consommateurs sont des mécanismes particulièrement intéressants du point de vue du *mechanism design* dans le cas où les biens échangés sont indiscernables. Cependant, dès lors que les biens sont individuellement différents de par leur nature ou simplement de par leur origine, ces protocoles ne sont plus adaptés. Dans le cas de notre plate-forme, chaque facture est différente, et son prix dépendra du nombre de jours restant avant le paiement, le donneur d'ordre, le fournisseur, le montant nominal etc. De plus, le protocole souffre des mêmes faiblesses que les enchères en ce qui concerne la liberté des agents. Le modèle producteur-consommateur est un protocole centralisé où les agents ne choisissent pas à qui leur bien sera vendu. Il est donc complexe d'y développer une relation acheteur-vendeur sur le long terme.

2.2.5 Discussion

Comme nous l'avons vu, le protocole répondant le mieux à notre problématique est le protocole de *bargaining*. En particulier, son implémentation multilatérale de négociation en parallèle permet aux agents d'intervenir, qu'ils soient acheteurs ou vendeurs. Ce protocole n'est cependant pas parfait, et il est nécessaire d'étudier les potentiels déséquilibres et effets néfastes que peut engendrer la divulgation d'information qui résulte de ce protocole.

La négociation peut être utilisée dans de nombreux contextes et à chacun d'entre eux répondra un ensemble de protocoles bien défini. Ainsi, dans les problèmes où des agents coopératifs utilisent la négociation pour se coordonner, certaines problématiques comme l'incitation se relâchent, et permettent la conception de protocoles spécialisés, par exemple pour l'allocation de tâches, comme le montrent [Smith, 1980, Belleili *et al.*, 2004, Choi *et al.*, 2009]. Dans ce contexte, les problématiques sont différentes : on recherchera à optimiser le temps de convergence dans l'allocation des ressources ou encore à effectuer cette allocation dans un temps imparti. Le contexte compétitif dans lequel se situe cette thèse nous pousse à nous intéresser à d'autres aspects, les agents cherchant parfois à manipuler le protocole pour obtenir une meilleure utilité aux dépens de leurs partenaires.

En utilisant le *mechanism design*, nous présentons une réponse à ces questions tout en conservant

un protocole symétrique, où les agents n'ont pas connaissance de l'utilité de leur partenaire. Ce domaine propose d'utiliser les outils mathématiques et une axiomatique sur les propriétés désirables d'un protocole afin de faire un choix éclairé. Il permet ainsi de comparer les protocoles en fonction de leurs propriétés, comme le fait [Sandholm, 1999] qui propose une analyse pratique basée sur l'utilisation de ces protocoles d'une part, et d'autre part une analyse théorique basée sur les axiomes de *mechanism design*.

2.3 *Mechanism design*

Dans la première partie de ce chapitre, nous nous sommes intéressés à un certain nombre de protocoles. Dans cette section, nous nous intéressons aux outils mathématiques permettant d'analyser et de concevoir des protocoles respectant certaines propriétés.

Cette manière d'analyser les jeux, en partant des comportements qu'ils incitent à suivre se nomme *Mechanism design* (en français, mécanismes d'incitation). Il est parfois appelée *inverse game theory* en ce qu'elle consiste à concevoir des jeux plutôt qu'à s'intéresser aux équilibres de jeux existants. Le travail de trois économistes ayant travaillé dans ce domaine, Leonid Hurwicz, Eric Maskin et Roger Myerson, s'est vu récompensé en 2007 par le Prix de la Banque de Suède en sciences économiques en mémoire d'Alfred Nobel, dit « Prix Nobel d'économie ».

Myerson en particulier s'est intéressé au problème du *bargaining* dans des conditions réalistes où l'acheteur et le vendeur n'ont pas la même information (en particulier, chacun ne connaît que son propre prix de réserve).

Dans cette section, nous nous concentrons d'abord sur les premiers résultats généraux de *mechanism design*, avant de focaliser davantage notre attention sur le protocole de *bargaining*, et les résultats obtenus par Meyerson sur ce protocole.

2.3.1 Principes

Avant de nous pencher sur le problème du *bargaining*, nous nous intéressons aux grands résultats de *mechanism design* [Hurwicz et Reiter, 2006]. Nous commençons par introduire le formalisme que nous utilisons et les définitions des propriétés souhaitables pour un mécanisme que nous avons évoqué en introduction de ce chapitre. Nous énonçons ensuite un certain nombre de résultats connus en *mechanism design* qui nous servirons dans la suite de ce chapitre. Le lecteur intéressé pourra se reporter à [Shoham et Leyton-Brown, 2008a]

2.3.1.1 Formalisme et définitions

Les principaux résultats de *mechanism design* concernent les mécanismes. Un mécanisme est défini comme une fonction qui à une action faite par un joueur associe une probabilité sur les possibles résultats d'un jeu bayésien. Il est donc nécessaire de définir au préalable ce qu'est un jeu bayésien. Afin de faciliter la lecture et la compréhension, nous définissons d'abord les jeux stratégiques.

Définition 2.3 (Jeu stratégique)
Un jeu stratégique est un quadruplet (N, Θ, Z, u) *où :*

- *N est l'ensemble des joueurs ; les joueurs sont représentés par des naturels de 1 à N,*
- *$\Theta = (\Theta_i)_{i \leq N}$, où chaque Θ_i est un vecteur représentant les types possibles de chaque joueur.*
- *Z est l'ensemble des résultats du jeu, i.e. les situations du jeu pour lesquelles on peut « calculer un score »,*
- *une fonction d'utilité pour chaque joueur $(u_i)_{i \leq N}$ définie sur $\mathbb{R}^{Z \times \Theta}$.*

Soit l'exemple de jeu stratégique suivant. Deux joueurs sont supposés coopérer pour accomplir une mission. Cependant, ces agents peuvent être des saboteurs dont le but est de faire échouer la mission. Ce jeu comporte deux joueurs ($N = 2$), et propose trois actions : coopérer c, faire défaut d ou agir seul a (notons que si les deux joueurs agissent seuls, les actions sont toutes répétées deux fois et la mission est réussie). Chaque agent peut être un travailleur, de type w ou un saboteur de type s. La notation θ_k^i indique que l'agent i est de type $k \in \{w, s\}$. Les résultats possibles sont "mission réussie" si les deux joueurs coopèrent et "mission échouée" si au moins un des agents fait défaut. La fonction d'utilité associe 1 à un joueur de type w ayant coopéré ou fait défaut, 0 si la mission a échoué (quelle que soit son action) et 0.5 si la mission a été réussie et que l'agent a accompli toutes les tâches. À l'inverse, un agent saboteur reçoit pour utilité 1 dans le cas où la mission échoue et -1 dans tous les autres cas. L'utilité des joueurs en fonction de leur type et de leurs actions sont données dans le Tableau 2.2.

Θ		θ_w^2			θ_s^2		
		c	d	a	c	d	a
θ_w^1	c	$(1,1)$	$(-1,-1)$	$(1,0.5)$	$(1,-1)$	$(-1,1)$	$(1,-1)$
	d	$(-1,-1)$	$(-1,-1)$	$(1,0.5)$	$(-1,1)$	$(-1,1)$	$(1,-1)$
	a	$(0.5,1)$	$(0.5,1)$	$(0.5,0.5)$	$(0.5,-1)$	$(0.5,-1)$	$(0.5,-1)$
θ_s^1	c	$(-1,1)$	$(1,-1)$	$(-1,0.5)$	$(-1,-1)$	$(1,1)$	$(-1,-1)$
	d	$(1,-1)$	$(1,-1)$	$(-1,0.5)$	$(1,1)$	$(1,1)$	$(-1,-1)$
	a	$(-1,1)$	$(-1,1)$	$(-1,0.5)$	$(-1,-1)$	$(-1,-1)$	$(-1,-1)$

TABLE 2.2 – Table des utilités du jeu de sabotage

Définition 2.4 (Jeu bayésien)
Un jeu bayésien (stratégique) est un quintuplet (N, Θ, Z, p, u) où p est une partition de probabilité connue a priori par tous les agents sur Θ. Cette probabilité définit ce qui est connu par les agents sur les stratégies adoptées par les autres agents

Un jeu bayésien associe à chaque type de joueur une probabilité *a priori*. Dans notre exemple, on peut supposer qu'on a une certaine connaissance des joueurs. On peut alors associer une probabilité pour chaque type : on peut par exemple supposer que, du fait de ses actions passées, on suppose que la probabilité que le joueur 1 soit un saboteur est de 25%, tandis que cette probabilité s'élève à 45% pour le joueur 2. Supposons maintenant que nous soyons le joueur 1. L'information apportée par l'aspect bayésien sur le jeu permet aux joueurs d'adapter leur stratégie. Si le joueur 1 est en fait de type w, le fait de savoir que le joueur 2 a une probabilité de 55% d'être un saboteur l'encourage à jouer l'action a plutôt que l'action c.

Le *mechanism design* s'intéresse aux protocoles en économie. L'un des aspects les plus utilisés pour créer un mécanisme qui a une stratégie dominante est d'ajouter une notion de paiement, de prix. Dans ce cas, les résultats possibles peuvent se décomposer en un résultat non monétaire et un résultat monétaire. Ce sont sur ces deux éléments que la fonction d'utilité sera définie.

Définition 2.5 (Fonction d'utilité quasi-linéaire)
Soit $B = (N, \Theta, Z, p, u)$ un jeu tel que $Z = X \times \mathbb{R}^N$. La fonction d'utilité u est dite quasi linéaire si et

seulement si la fonction u est décomposable :

$$\forall\theta \in \Theta, \forall i \leq N, \forall z = (x, (\pi_i)_{i\leq N}) \in Z, \exists u'_i \in \mathbb{R}^X, f_i \in \mathbb{R}^X : u_i(z,\theta) = u'_i(x,\theta) - f_i(\pi_i,\theta) \tag{2.8}$$

L'Équation (2.8) illustre que la fonction d'utilité se décompose en la somme d'une fonction qui dépend du résultat non monétaire (par exemple le résultat d'une négociation, ou d'un vote) et d'une autre qui dépend uniquement du prix reçu par le joueur. Notons enfin que le signe devant la fonction de paiement est négatif car on considère généralement qu'il s'agit du prix à payer. La fonction f_i fait quant à elle l'association entre l'argent reçu et la fonction d'utilité de l'agent. Si on peut supposer qu'elle est croissante, rien n'assure qu'elle est linéaire.

Le fait d'ajouter un prix se traduit dans l'ensemble des résultats possibles, qui se décompose en un produit cartésien de résultats non monétaires dans un ensemble nommé X et d'un ensemble de prix (un par agent) appartenant donc à $\mathbb{R}^N$: $Z = X \times \mathbb{R}^N$. Si tel est le cas, il est également possible de réaliser cette séparation sur le mécanisme. On peut alors définir un mécanisme quasi-linéaire.

Définition 2.6 (Mécanisme)
Un mécanisme sur un jeu bayésien $B = (N, \Theta, Z, p, u)$ est un couple (A, M) défini par :

- $A = \prod_{k\leq N} (A_i)$*, où A_i est l'ensemble des actions que i peut faire,*
- $M \in \Pi(Z)^A$ *est une fonction qui associe à chaque action une distribution de probabilité* Π *sur les différents résultats possibles.*

Définition 2.7 (Mécanisme quasi-linéaire)
Un mécanisme quasi linéaire μ sur un jeu $B = (N, \Theta, Z, p, u)$ tel que $Z = X \times \mathbb{R}^N$ est un triplet $\mu = (A, m, \wp)$ tel que :

- $A = \prod_{k\leq N} (A_i)$*, où A_i est l'ensemble des actions que i peut faire,*
- $m \in \Pi(X)^A$ *est une fonction qui associe à chaque action une distribution de probabilité* Π *sur les différents résultats non monétaires possibles,*
- $\wp : A \to \mathbb{R}^N$ *est une fonction qui, pour un ensemble d'actions faites par tous les agents, associe à chacun un prix à payer.*

Dans le cas où le résultat non monétaire ne dépend que de l'action faite par un agent, le mécanisme est dit déterministe

Définition 2.8 (Mécanisme déterministe)
Un mécanisme $\mu = (A, M)$ sur un jeu bayésien $B = (N, \Theta, Z, p, u)$ est dit déterministe si et seulement si

$$\forall a \in A, \exists z \in Z : m(a)(z) = 1 \tag{2.9}$$

Pour un mécanisme déterministe $\mu = (A, M)$, on note indifféremment $M(A) = z \in Z : m(a)(z) = 1$

En *mechanism design*, l'action réalisée par un joueur est souvent d'annoncer son type. C'est par exemple le cas dans les enchères de Vickrey, où le prix de réserve d'un agent correspond précisément à cette notion. Dans ce cas, on dit que le mécanisme est direct.

Définition 2.9 (Mécanisme direct)
Soit $B = (N, \Theta, Z, p, u)$ et $\mu = (A, M)$ un mécanisme sur B. μ est dit direct si et seulement si $A = \Theta$

Les mécanismes dits de Groves, ont attiré l'attention en raison de leurs propriétés (voir le Théorème 2.13). Ils reposent notamment sur la notion de stratégie dominante.

Définition 2.10 (Stratégie dominante)
Une stratégie dominante pour un joueur i dans un mécanisme $\mu = (A, M)$ sur un jeu $B = (N, \Theta, Z, p, u)$ est une action $a^* \in A_i$ telle que

$$\forall \theta \in \Theta_i, \forall a \in A_i, \sum_{z \in Z} m(a)(x) \cdot u_i(\theta, z) \leq \sum_{z \in Z} m(a^*)(z) \cdot u_i(\theta, z) \tag{2.10}$$

D'après l'Équation (2.10), quel que soit le profil du joueur, l'action a^* lui donne une plus grande utilité moyenne que n'importe quelle autre action.

Définition 2.11 (Mécanisme de Groves)
Soit $B = (N, \Theta, Z, p, u)$un jeu bayésien dont l'utilité est quasi linéaire (avec les mêmes notations que dans la Définition 2.5). Soit $\mu = (A, m, \wp)$ un mécanisme quasi-linéaire direct et déterministe sur B. Alors μ est un mécanisme de Groves si et seulement si la stratégie dominante a^* vérifie :

- $m(a^*) = \underset{x \in X}{\operatorname{argmax}} \left(\sum_{i \leq N} u'_i(x) \right)$
- $\exists (h_i)_{i \leq N} \in \left(\mathbb{R}^{N-1}\right)^{\mathbb{R}} : \wp_i(a^*) = h_i((p_k)_{k \neq i}) - \sum_{j \neq i} u'_j(m(a^*))$

Le mécanisme de Groves implémente ainsi le bien être social utilitariste, au sens de la Définition 2.12 et le prix payé par chaque agent ne dépend que des utilités des autres agents (et non de la sienne). L'enchère de Vickrey, décrite en Section 2.2.1.1, est un exemple de mécanisme de Groves.

Ces définitions constituent les notions de base en *mechanism design*. Le but est ensuite d'utiliser ces concepts afin de définir des mécanismes ayant certaines propriétés. Nous donnons maintenant la définition formelle de ces propriétés qui sont décrites plus en détail par [Shoham et Leyton-Brown, 2008a, Sandholm, 1999, Kraus, 2001].

La première caractéristique désirable concerne les fonctions de classement de bien-être social. Une fonction de classement de bien-être social permet de comparer des situations (typiquement dans notre cas, des résultats) selon un critère dépendant de l'utilité.

Définition 2.12 (Implémentation du bien être social)
Soit $\prec$ une fonction de classement sur Z^2. Un mécanisme déterministe $\mu = (A, M)$ implémente $\prec$ si et seulement si il existe $a^* = (a_i^*)_{i \in [\![1,N]\!]} \in A$ tel que a_i^* est une stratégie dominante pour i et

$$\forall a \in A, M(a) \prec M(a^*) \tag{2.11}$$

L'une des caractéristiques majeures pour un mécanisme est la stabilité. Un mécanisme direct est incitatif si et seulement si une stratégie dominante est de révéler son type.

Définition 2.13 (Mécanisme incitatif)
Soit $\mu = (A = \Theta, M)$ un mécanisme direct. μ est dit incitatif si et seulement si θ_i est une stratégie dominante.

L'un des aspects les plus fondamentaux en théorie des jeux est l'optimalité de Pareto. Un optimum de Pareto est un résultat tel que tout autre résultat implique que l'un des agents au moins voit son utilité décroître. L'optimum de Pareto est une situation optimale au sens où un changement implique forcément qu'un des agents perd au change. Cela ne signifie pas que le résultat est juste ou désirable. Dans une négociation, une situation où un des agents reçoit une utilité nulle tandis que l'autre reçoit

une utilité très élevée peut être Pareto optimale, si toute augmentation de l'utilité de l'agent recevant une utilité nulle entraîne une baisse de l'utilité de l'autre agent. Un mécanisme est dit Pareto-efficace s'il existe une stratégie dominante pour chaque joueur telle que l'exécution de ces stratégies par les agents mène à un résultat Pareto-optimal.

Définition 2.14 (Efficacité de Pareto)
Un mécanisme $\mu = (A, m, \wp)$ est dit Pareto-efficace si et seulement si il existe $A^* = (a_i^*)_{i \leq N}$ tel que a_i* est une stratégie dominante pour l'agent i.

Un mécanisme respecte la rationalité individuelle si et seulement si pour tout agent, il est plus intéressant de participer au mécanisme (et d'utiliser une stratégie dominante) que de ne pas y participer.

Définition 2.15 (Rationalité individuelle)
Soit μ un mécanisme déterministe. μ respecte la rationalité individuelle si et seulement s'il existe $A^* = (a_i^*)_{i \leq N}$ tel que a_i* est une stratégie dominante pour l'agent i et tel que $u(M(a^*)) \geq 0$.

Une dernière propriété désirable pour des protocoles est la profitabilité. Un mécanisme est dit profitable si et seulement si le prix payé par l'ensemble des agents est positif ou nul :

Définition 2.16 (Mécanisme profitable)
Soit $\mu = (A, m, \wp)$ un mécanisme quasi-linéaire et a^* une stratégie dominante. μ est profitable si et seulement si $\sum_{i \leq N} \wp_i(a^*) \geq 0$.

Un mécanisme est dit équilibré si le prix payé par l'ensemble des agents est nul :

Définition 2.17 (Mécanisme équilibré)
Soit $\mu = (A, m, \wp)$ un mécanisme quasi-linéaire et a^* une stratégie dominante. μ est équilibré si et seulement si $\sum_{i \leq N} \wp_i(a^*) = 0$.

Au-delà des propriétés souhaitables pour un protocole, il existe un certain nombre de propriétés non désirables pour les protocoles. Un exemple en est la propriété de dictature.

Définition 2.18 (Dictature)
Un mécanisme μ est une dictature si et seulement si il existe $i \leq N$ tel que le μ implémente la fonction de choix social $\prec_i$ suivante $\forall (z, z') \in Z^2, z \prec_i z' \iff u_i(z) \leq u_i(z')$.

Les autres propriétés que nous avons données concernent des aspects d'implémentation de ces protocoles, et nous ne les décrirons pas ici. Ces définitions permettront d'énoncer les théorèmes de *mechanism design* ayant un rapport avec notre place de marché. Elles nous permettront également de démontrer un certain nombre de propriétés sur le *bargaining*.

2.3.1.2 Résultats de mechanism design

Cette section introduit un certain nombre de résultats généraux de *mechanism design*. Ces résultats seront mis en relation avec le protocole de *bargaining* dans la Section 2.3.2.

Le premier théorème est nommé principe de révélation, et donne une correspondance entre un mécanisme implémentant une fonction de choix social et un mécanisme direct incitatif implémentant cette même fonction. Il a été démontré pour la première fois par [Gibbard, 1973].

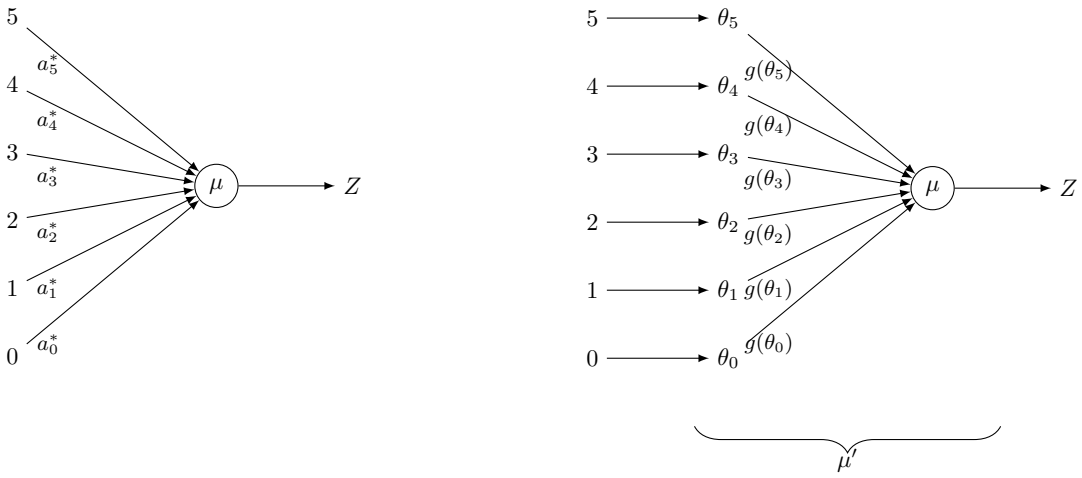
Théorème 2.12 (Principe de révélation)
Si un mécanisme μ implémente une fonction de bien-être social $\prec$, alors, il existe un mécanisme direct incitatif μ' implémentant aussi $\prec$.

Démonstration. Soit $\mu = (A, m, \wp)$ le mécanisme sur B. Soit a^* une stratégie dominante de B. On construit g, une fonction associant à chaque type d'agents sa stratégie dominante.

$$\begin{aligned} g :& \Theta \to A \\ & (\theta^i)_{i \leq N} \mapsto a_i^* \end{aligned} \tag{2.12}$$

Le mécanisme $\mu' = (\Theta, M')$ où $M' = g \circ M$ est un mécanisme direct. Montrons par l'absurde que ce mécanisme est incitatif. Supposons qu'il existe pour un joueur i une stratégie qui soit « meilleure » que déclarer son propre type (par exemple θ_0), c'est à dire que cette action amène plus d'argent. Alors, notons $a_0 = g(\theta_0)$. Puisque la stratégie de présenter θ_0 est meilleure que θ_i, on a $u_i(M(a_0)) > u_i(M(g(\theta_i)))$. Or, $g(\theta_i) = a_i^*$ donc $u_i(M(a_0)) > u_i(M(a_i^*))$ ce qui est contradictoire avec la définition d'une stratégie dominante. □

Une illustration de ce principe de révélation est donnée en Figure 2.5.



(a) Mécanisme implémentant une fonction de bien-être social

(b) Mécanisme direct incitatif correspondant

FIGURE 2.5 – Principe de révélation

Théorème 2.13 (Stabilité des mécanismes de Groves)
Les mécanismes de Groves sont incitatifs.

Les mécanismes de Groves ont été décrits par [Groves, 1973]. Le lecteur intéressé par la démonstration du théorème pourra s'y référer.

2.3.2 *Mechanism design* et *bargaining*

Nous nous sommes jusqu'à présent intéressés aux mécanismes en général. Ces mécanismes sont applicables à tous types de protocole, allant des procédures de vote aux enchères. Dans cette partie,

nous nous concentrons sur l'application des outils que nous avons décrits au *bargaining*.

2.3.2.1 Information complète

La première analyse du *bargaining* a été réalisée par [Nash, 1950]. Pour Nash, le *bargaining* est une situation dans laquelle deux agents tentent de trouver un accord. Les agents ont la possibilité de trouver un accord intéressant pour les deux, *i.e.* une situation dans laquelle leur utilité est positive, mais ils ne sont pas d'accord sur l'accord à réaliser. Nash considère une situation où l'information est complète, *i.e.* les deux agents connaissent le prix de réserve l'un de l'autre. Par ailleurs, il considère un *bargaining* mono-attribut dont l'attribut est continu. Une définition plus formelle est donnée à la Définition 2.19

Définition 2.19 (*Bargaining* de Nash)
Le bargaining est une paire (F, v) où F représente l'ensemble des accords réalisables $F \subset \mathbb{R}^2$, et $v \in \mathbb{R}^2$ est l'ensemble des prix de réserve des agents.

Afin qu'il y ait une solution au problème, il est donc nécessaire que $F \cup \{(x_1, x_2) \in \mathbb{R}^2 | x_1 \geq v_1 \wedge x_2 \geq v_2\} \neq \varnothing$.

F est d'autre part supposé convexe et fermé. Supposons la situation suivante : deux agents doivent se retrouver pour partager 10€. Ils ont la possibilité de se partager l'intégralité de la somme, ou d'en laisser une partie à la personne qui doit les leur remettre. Dans tous les cas, ils doivent se retrouver loin de chez eux : le voyage coûte 2€à 1 ou 3€à 2. La Figure 2.6 représente la situation. On note $A = \{(x_1, x_2) \in \mathbb{R}^2 | x_1 \geq v_1 \wedge x_2 \geq v_2\}$ l'ensemble des solutions acceptables par les deux agents.

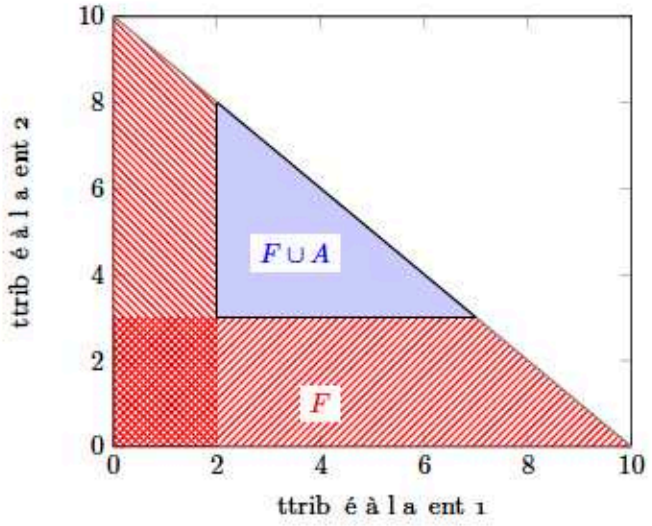


FIGURE 2.6 – Espace réalisable et espace acceptable

Dans ce contexte, Nash se fixe comme objectif de trouver une solution dans F. Conceptuellement, tout couple dans $F \cup A$ est une solution acceptable pour le problème du bargaining. [Nash, 1950] propose donc une approche axiomatique qui repose sur 5 axiomes :

Efficacité de Pareto Si (x_1^*, x_2^*) est une solution, alors $\forall (x_1, x_2) \in F \cup A, x_1 \leq x_1^* \vee x_2 \leq x_2^*$ (identique à la Définition 2.14)

Rationalité individuelle Si (x_1^*, x_2^*) est une solution au problème, alors $x_1^* > v_1 \wedge x_2^* > v_2$ (identique à la Définition 2.15)

Indépendance aux alternatives non pertinentes Soit $G \subset F$ et (x_1^*, x_2^*) une solution de F. Alors, si $(x_1^*, x_2^*) \in G$, (x_1^*, x_2^*) est aussi une solution de G.

Symétrie Si $v_1 = v_2$ et $(x_1, x_2) \in F \iff (x_2, x_1) \in F$, alors si (x_1^*, x_2^*) est une solution de F, $x_1^* = x_2^*$.

Linéarité soit $G = \{(\lambda_1 \cdot x_1 + \mu_1, \lambda_2 \cdot x_2 + \mu_2) | (x_1, x_2) \in F\}$ avec $(\lambda_1, \lambda_2, \mu_1, \mu_2) \in \mathbb{R}^4$, $(w_1, w_2) = (\lambda_1 \cdot v_1 + \mu_1, \lambda_2 \cdot v_2 + \mu_2)$ et (x_1^*, x_2^*) une solution de (F, v) alors la solution de (G, w) est $(\lambda_1 \cdot x_1^* + \mu_1, \lambda_2 \cdot x_2^* + \mu_2)$.

Cette propriété peut se traduire ainsi : étant donnée une solution pour un problème, une solution pour le même problème après une transformation affine est la transformation affine de la solution initiale.

Étant donnés ces axiomes, [Nash, 1950] démontre le Théorème 2.14.

Théorème 2.14 (Existence et unicité de la solution de Nash)
Soit (F, v) un problème de bargaining. Il existe une et une seule solution vérifiant les axiomes d'efficacité de Pareto, de rationalité individuelle, de linéarité, d'indépendance aux alternatives non pertinentes et de symétrie. Cette solution, (x_1^*, x_2^*) vérifie en outre :

$$(x_1^*, x_2^*) = \underset{(x_1, x_2) \in F \cup A}{\operatorname{argmax}} (x_1 - v_1) \cdot (x_2 - v_2) \tag{2.13}$$

2.3.2.2 Information incomplète

[Myerson, 1979] reprend l'analyse de Nash, mais dans un contexte où l'information est imparfaite. Dans ce contexte, les agents cherchent à partager un bien divisible. Cependant, les types des agents ne sont pas connus.

Une des particularités du *bargaining* dans les travaux de Myerson est qu'elle porte sur une version comportant un médiateur (ou *broker*). Dans cette version, les agents ne négocient pas directement l'un avec l'autre mais confient leurs préférences à un tiers de confiance, qui en déduit une solution. Ce protocole est avantageux puisqu'il permet aux agents de profiter de la connaissance de leurs informations secrètes respectives sans que leurs préférences soient révélées.

De ce contexte, il est possible de construire un jeu. Ce jeu a N joueurs, le type des joueurs est constitué des prix qu'ils sont prêts à payer (leurs prix de réserve), un résultat du jeu est un élément de F et les agents sont dotés d'une fonction d'utilité en fonction de l'élément de F choisi.

On peut alors déduire de ce contexte un mécanisme direct selon le Théorème 2.12, où les agents déclarent leur type, et où une solution – un vecteur correspondant à ce que chacun reçoit – est choisie à partir de ces types. En *bargaining*, le type d'un agent est son prix de réserve, le prix maximal qu'il accepte de payer ou le prix minimal qu'il accepte de recevoir (v dans la notation utilisée jusqu'à présent). Dans ce cadre, Myerson cherche à trouver une solution au problème du *bargaining* qui soit Pareto-optimale. Il montre que dans le cas général, seule la dictature est une solution de ce type.

Ce résultat est développé par [Myerson et Satterthwaite, 1983], qui démontre le Théorème 2.15 :

Théorème 2.15 (Théorème de Myerson-Satterthwaite)
Il n'existe pas de solution au bargaining tel que défini ci-dessus qui soit à la fois incitatif, profitable, Pareto-efficace et qui respecte la rationalité individuelle.

Ce théorème reste valable même dans le cas où les agents ont une préférence quasi-linéaire.

Si le théorème ne peut être remis en question, les axiomes sur lesquels il repose peuvent l'être. Myerson s'attaque à l'axiome de Pareto-efficacité. En effet, l'une des suppositions faites par le *bargaining* est que l'accord n'a lieu que si les deux agents donnent leur accord. Ainsi, chercher un optimal de Pareto dans l'ensemble de F n'a pas de sens, y compris dans l'espace où la condition de rationalité individuelle puisque les agents ne donneraient pas un accord à une solution si jamais elle ne correspondait pas à une stratégie dominante.

Pour répondre à cette limitation, Myerson propose de restreindre la recherche à l'ensemble $F^* \subset F$ des assignations respectant la rationalité individuelle, c'est à dire de créer un mécanisme incitatif. Il montre ainsi que si un optimal de Pareto est recherché sur $F^* \cup A$, une unique solution existe qui maximise une agrégation particulière des utilités des agents (par exemple leur produit).

Théorème 2.16 (Existence et unicité de la solution au problème du *bargaining* à information incomplète dans F^*)
Si v n'est pas Pareto-efficace, il existe une et une seule solution au problème du bargaining qui soit incitatif, profitable, Pareto-efficace dans F^*, qui respecte la rationalité individuelle et maximise le produit de Nash.

Afin de donner un meilleur aperçu de ces résultats, nous considérons l'exemple suivant : deux industriels veulent cotiser pour acheter une machine. L'un des industriels a véritablement besoin de cette machine : il estime qu'elle lui ferait gagner 5.000€. Pour l'autre industriel, c'est plus compliqué. Il a accès à des services équivalents à la machine, et il prétend que l'achat de la machine lui ferait gagner 2.000€, mais le prix de ces services sur le marché est plus cher que ce qu'il prétend, et s'il paie les services au prix du marché, la machine lui ferait également gagner 5.000€. Il est impossible de savoir s'il ment ou non, mais on peut supposer qu'il est assez probable qu'il paie son service au prix du marché. On peut donc considérer que le type de 1 est connu et est 5. Le type de 2 est lui plus compliqué. Il est composé de $\theta_2^1 = 2$ et de $\theta_2^2 = 5$. De plus, les connaissances *a priori* sur le type de 2 nous donnent : $p(\theta_2^1) = 0.25$ et $p(\theta_2^2) = 0.75$. La machine coûte 6.000€ ; il est donc intéressant de l'acheter pour les agents dans tous les cas.

Les résultats possibles sont les suivants :

1. les agents n'achètent pas la machine $z_0 = (0, 0)$. Dans ce cas, les agents obtiennent une utilité nulle $\forall \theta \in \Theta u(\theta, z_0) = (0, 0)$,
2. 1 paie l'intégralité de la machine. Dans ce cas, l'utilité de 2 dépend de son type : $u(\theta_1) = -1$ et $u_1(\theta_1) = -1$ tandis que $(u_2(\theta_2^1) = 2)$ et $(u_2(\theta_2^2) = 5)$,
3. enfin, si 2 paie l'intégralité de la machine : $u_1(\theta_1) = 5$ tandis que $(u_2(\theta_2^1) = -4)$ et $(u_2(\theta_2^2) = -1)$.

Il existe bien entendu toute une série d'allocations intermédiaires. Cependant, ces résultats sont équivalents à des implémentations probabilistes des solutions ci-dessus. Par exemple, un cas où l'agent 1 paie 3 et l'agent 2 paie 3 est en fait la moyenne des cas 2 et 3, et peut être vu comme un mécanisme qui sélectionne la solution 2 50% du temps et la solution 3 50% du temps. La solution 1 est Pareto dominée par toute solution telle que l'utilité attendue des joueurs est positive ou nulle.

Le mécanisme le plus simple est de demander à chacun de déclarer son type et de répartir les coûts de manière proportionnelle au type annoncé. Les résultats sont donnés dans la Tableau 2.3. Pour alléger la notation, le type déclaré par l'agent 2 est noté $\hat{\theta}_2$. Comme on peut le voir, déclarer θ_2^1 est avantageux pour l'agent 2 et le protocole n'est donc pas incitatif.

Une autre solution est de choisir un dictateur. Comme nous l'avons exposé, si un des agents paie de toute façon l'ensemble du prix, alors le protocole est incitatif : si l'agent paie la même somme quoi

$\hat{\theta}_2$	θ_2^1	θ_2^2
$\theta_2 = \theta_2^1$	(0.7,0.3)	(2,-1)
$\theta_2 = \theta_2^2$	(0.7,3.3)	(2,2)

TABLE 2.3 – Utilité des agents en fonction du type de 2 et de son action

qu'il dise, il n'a pas d'intérêt à mentir. Le problème est qu'utiliser ce protocole paraît particulièrement injuste. En outre, le protocole ne respecte pas la rationalité individuelle : l'agent qui paie recevra une utilité négative.

Myerson propose une solution à ce problème sous la forme d'un mécanisme non déterministe, et qui n'est pas Pareto optimal dans F, mais qui l'est dans F^*. Pour ce faire, il propose de formuler le problème sous la forme d'un problème d'optimisation sous contrainte. On note les solutions :

0. la machine n'est pas achetée. 1 et 2 reçoivent chacun une utilité de 0,
1. la machine est achetée et 1 en paie l'intégralité. 2 reçoit une utilité de 5000,
2. la machine est achetée et 2 en paie l'intégralité. 2 reçoit une utilité de -1000.

L'utilité de 1 dépend de son type. Elle est résumée dans la Tableau 2.4.

solution	θ_2^1	θ_2^2
1	-4000	-1000
2	2000	5000

TABLE 2.4 – Utilité de l'agent 2 en fonction de la solution retenue

On note $p(i|j)$ la probabilité attribuée au prix i sachant que l'agent 2 a déclaré le type θ_2^j. L'utilité moyenne de 2, sachant qu'il est de type θ_2^1 et qu'il a déclaré $\hat{\theta_2^j}$ est :

$$p(0|j) \cdot (0) + p(1|j) \cdot (-4000) + p(2|j) \cdot (2000) \tag{2.14}$$

Son utilité moyenne s'il est de type θ_2^2 est

$$p(0|j) \cdot (0) + p(1|j) \cdot (-1000) + p(2|j) \cdot (5000) \tag{2.15}$$

La traduction de la stabilité se traduit par :

$$\begin{aligned} &p(0|1) \cdot (0) + p(1|1) \cdot (-4000) + p(2|1) \cdot (2000) \geq p(0|2) \cdot (0) + p(1|2) \cdot (-4000) + p(2|2) \cdot (2000) \\ &p(0|1) \cdot (0) + p(1|1) \cdot (-1000) + p(2|1) \cdot (5000) \leq p(0|2) \cdot (0) + p(1|2) \cdot (-1000) + p(2|2) \cdot (5000) \end{aligned} \tag{2.16}$$

De plus, les probabilités de choix de chaque solution dans le cas où 2 déclare chaque type doivent se sommer à 1 :

$$\begin{aligned} p(0|1) + p(1|1) + p(2|1) = 1 \\ p(0|2) + p(1|2) + p(2|2) = 1 \end{aligned} \tag{2.17}$$

On peut alors noter l'utilité moyenne de 1 u_1, de 2 sachant qu'il est de type θ_2^1, $u_{2.1}$, et de 2 sachant qu'il est de type θ_2^2, $u_{2.2}$ l'utilité de 2 sachant qu'il est de type θ_2^2. En supposant que l'agent 2 dit vrai :

$$\begin{aligned} u_1 = \quad & 0.25 \cdot (p(0|1) \cdot (0) + p(1|1) \cdot (-1000) + p(2|1) \cdot (5000)) \\ & +0.75 \cdot (p(0|2) \cdot (0) + p(1|2) \cdot (-1000) + p(2|2) \cdot (5000)) \\ u_{2.1} = \quad & p(0|1) \cdot (0) + p(1|1) \cdot (2000) + p(2|1) \cdot (-4000) \\ u_{2.1} = \quad & p(0|2) \cdot (0) + p(1|2) \cdot (5000) + p(2|2) \cdot (-1000) \end{aligned} \tag{2.18}$$

Il reste à définir l'optimal de Pareto. [Myerson, 1979] propose de trouver un optimal en particulier, l'optimal au sens du produit de Nash. Cela lui permet de formuler la recherche d'un optimal de Pareto comme une optimisation sous contrainte, où la fonction objectif est le produit de Nash. Mais cette fonction n'est pas la seule qui est Pareto efficace (au sens où le maximum de cette fonction est un optimum de Pareto). C'est aussi notamment le cas de la fonction de bien-être utilitaire :

$$u_1 \cdot u_{2.1} \cdot u_{2.2} \tag{2.19}$$

La recherche d'un optimum de Pareto devient donc un problème d'optimisation :

optimiser	Équation (2.19)
sous les contraintes de	Équation (2.16)
	Équation (2.17)
	Équation (2.18)

Notons que ce problème est un problème d'optimisation non linéaire dans un ensemble convexe, représenté sur la Figure 2.7.

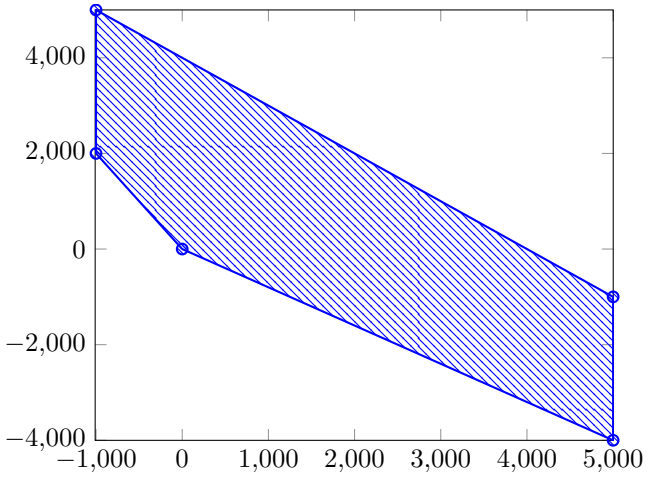


FIGURE 2.7 – Espace convexe d'optimisation

La solution que nous obtenons [4] est

$$\begin{aligned} p(0|1) &= 0.57 \\ p(1|1) &= 0.43 \\ p(2|1) &= 0 \\ p(0|2) &= 0 \\ p(1|2) &= 0.62 \\ p(2|2) &= 0.38 \\ u_1 &= 854.88 \\ u_{2,1} &= 858.54 \\ u_{2,2} &= 2717.1 \end{aligned} \tag{2.20}$$

Cette solution est pas optimale au sens de Pareto dans F. En effet, elle est Pareto dominée par la solution

$$\begin{aligned} p(0|1) &= 0 \\ p(1|1) &= 0.85 \\ p(2|1) &= 0.12 \\ p(0|2) &= 0 \\ p(1|2) &= 0.47 \\ p(2|2) &= 0.53 \\ u_1 &= 935 \\ u_{2,1} &= 1100 \\ u_{2,2} &= 2720 \end{aligned} \tag{2.21}$$

En revanche, cette dernière n'est dans pas dans F^*. La solution donnée dans l'Équation (2.20) est quant à elle l'optimum de la fonction objectif – et donc Pareto optimale – dans F^*.

2.3.3 Discussion

La variante du *bargaining* développée par Myerson est intéressante mais elle repose sur des axiomes qui ne font pas nécessairement l'unanimité : il existe plusieurs solutions dans l'espace F^* et on peut supposer que certains agents refuseront de maximiser cette fonction. On en revient alors au *bargaining* standard, le protocole que nous avons décrit dans la Section 2.2.2. Comme nous l'avons évoqué, ce protocole est cependant problématique, puisqu'il demande aux agents d'agir de manière complexe pour s'approcher autant que possible des solutions optimales réalisables maximisant leurs utilités respectives. C'est de cette complexité que traite le Chapitre 3.

Les techniques permettant d'arriver à un tel résultat impliquent d'acquérir des informations sur l'adversaire. Ces informations sont généralement considérées comme privées par ce dernier. Comme le fait remarquer [Myerson, 1991], « En général, chaque individu doit chercher à apprendre sur ses adversaires, et doit essayer d'influencer ce qu'ils croient » [5]. Si cela n'est pas problématique en général, un agent qui prend cette incitation au pied de la lettre pourrait tenter d'apprendre sur son adversaire, quitte à faire échouer la négociation. Pire, il pourrait démarrer une négociation sans intention de

4. Pour obtenir cette solution, nous utilisons le solveur d'Octave [Oct, 2017]
5. *In general, each individual must try to learn about his opponents and must try to influence what they believe*

trouver un accord, uniquement pour obtenir de l'information sur son adversaire. Ces comportements peuvent nuire au système, et c'est pourquoi nous proposons d'utiliser les mécanismes d'incitation tels que proposés dans ce chapitre pour faire en sorte qu'ils ne soient pas avantageux pour un agent.

2.4 Conclusion

Dans ce chapitre, nous avons présenté les principaux protocoles de négociation et des outils pour les analyser. Nous nous sommes particulièrement intéressés aux protocoles de *bargaining* parce qu'ils permettent aux deux agents d'exprimer leurs préférences dans un contexte où le domaine de négociation est complexe. Les variantes du *bargaining* sont nombreuses et permettent de confier l'aspect négociation à proprement parler du protocole à un tiers qui trouvera une solution optimale au regard de certains axiomes prédéterminés. Cependant, rien ne dit que les agents soient d'accord sur ces axiomes, notamment sur l'optimisation des critères de bien-être social. Dans ce cas, il est nécessaire de revenir à une version plus simple du *bargaining*, et de laisser les agents chercher par un échange de propositions un accord qui les satisfasse tous les deux. Dans cette configuration, l'apprentissage des préférences de l'adversaire redevient central afin de trouver une solution qui soit efficace. Il n'en reste pas moins que ces préférences sont privées et que le *bargaining* incite à les déduire des propositions reçues, ce qui peut poser problème.

Chapitre 3

Agents négociateurs

“
As Cut-me-own-Throat Dibbler knew in his heart, wherever two or more people are gathered together, someone will be trying to sell them a suspicious sausage in a bun.
Terry PRATCHETT, *Moving pictures*, 1990

”

Sommaire

3.1 Introduction 68
3.2 Modélisation de l'adversaire 69
3.2.1 Stratégie d'offres 70
3.2.2 Le prix de réserve 74
3.2.3 La fonction d'utilité 75
3.2.4 Stratégie d'acceptation 77
3.2.5 Deadline 78
3.2.6 Discussion 79
3.3 Stratégies d'offres 79
3.3.1 Stratégies fixes 79
3.3.2 Stratégies adaptatives utilisant des stratégies fixes 83
3.3.3 Intervalles 85
3.3.4 Algorithmes génétiques 85
3.3.5 En utilisant les propositions de l'adversaire 87
3.3.6 Discussion 87
3.4 Stratégie d'acceptation 88
3.4.1 Stratégies simples 88
3.4.2 Combinaison de stratégies simples 89
3.4.3 Stratégie optimale 89
3.4.4 Courbe de concession 89
3.4.5 Discussion 90
3.5 Conclusion 90

3.1 Introduction

La négociation tient une place centrale dans nos vie. Nous négocions en permanence : avec notre famille et nos amis par exemple, avec des marchands, physiquement ou non, avec notre banque. Comme nous l'avons expliqué dans le Chapitre 2, la négociation est aussi un mécanisme de coordination, qui peut servir à allouer des ressources dans un système (voir par exemple [An *et al.*, 2010, Mailler *et al.*, 2003]). Si toutes les négociations ne sont pas bonnes à automatiser (on s'imagine mal confier les négociations que l'on fait au quotidien au sein de la famille à une intelligence artificielle), il est des cas ou il peut être opportun de laisser des agents négocier au nom de leur utilisateur. Notre place de marché permet de faire de la négociation de créance. Dans ce cadre, confier la négociation à un agent automatique peut permettre d'obtenir de meilleurs résultats (un agent logiciel peut explorer un espace de négociation large plus rapidement et plus efficacement) et donc d'éviter au négociateur humain de répéter des négociations similaires avec les mêmes partenaires, tout en permettant d'économiser le temps des collaborateurs.

Notre place de marché met donc à disposition de ses clients, fournisseurs comme investisseurs, un agent de négociation automatique. Cet agent doit être autonome et rationnel. En particulier, étant donné un protocole et un profil de préférence faisant correspondre une fonction d'utilité à chaque agent, celui-ci doit être capable de traiter les informations qu'il reçoit et d'agir en conséquence. Son but est de maximiser l'utilité qui lui est attribué. Dans la pratique, un tel agent est spécifiquement conçu pour un protocole donné. A notre connaissance, le protocole ayant été le plus largement étudié dans la litterature est le *bargaining*, que nous avons introduit en Section 2.2.2.

En particulier depuis 2009, une compétition annuelle de négociation automatique (ANAC pour *Automated Negotiation Agent Competition* [ANA, 2011, Baarslag *et al.*, 2015a]) a été lancée. Elle s'est concentrée dans ses cinq premières éditions sur le *bargaining*, relâchant peu à peu des contraintes afin de s'approcher d'une négociation en situation réelle. En 2014 par exemple, les fonctions d'utilité des agents n'étaient plus linéaires contrairement aux années précédentes. Les compétitions suivantes se sont focalisées sur de nouveaux protocoles. En 2015, le protocole utilisé par la compétition était une extension du *bargaining* à un contexte trilatéral, où trois agents devaient se mettre d'accord pour que la négociation aboutisse. En 2016, la compétition s'est concentrée sur un contexte de *smart grids*. La compétition de 2017 a quant à elle proposé plusieurs catégories : une catégorie pour le *bargaining*, une basée sur le jeu Diplomatie [Sharp, 1978], et une face à des adversaires humains.

Il existe de nombreux modèles d'agents de négociation, très différents. Il est donc complexe de les analyser de manière cohérente. Leur fonctionnement varie énormément, et tandis que certains agents se concentrent sur la recherche de propositions acceptables pour eux-mêmes et pour l'adversaire, d'autres cherchent plutôt à optimiser leur utilité, aux dépens de celle de leur adversaire, tout en tentant d'éviter un rejet de sa part.

Il est cependant possible de trouver entre tous ces agents certains composants identiques. Dans ce chapitre, nous adoptons le formalisme introduit par [Baarslag, 2016]. Ce formalisme se nomme BOA pour *Bidding strategy*, *Opponent modeling* et *Acceptance Strategy*, et considère que les agents de négociation peuvent se diviser en trois grands composants : une stratégie d'offres, un modèle de l'adversaire [1] et une stratégie décidant s'il faut accepter ou non une proposition reçue. Cette formalisation a été conçue pour le *bargaining* et n'est donc pas adaptée à tous les protocoles. Dans le cas de protocoles ayant des stratégies dominantes en particulier (telle que l'enchère de Vickrey décrite en Section 2.2.1.1),

1. Dans toute la suite de ce manuscrit, on appellera le ou les agents avec lesquels un agent négocie son ou ses « adversaire(s) ». Ce terme est une traduction du terme anglais *opponent*, présent dans l'expression *opponent modeling* et qui a d'abord été utilisé dans les jeux.

la génération d'offres se résume à adopter cette stratégie dominante. De même, dans le cadre d'un protocole à un tour non répété, il est impossible de modéliser l'adversaire. Enfin, la stratégie d'acceptation n'a pas de sens que dans le cas où l'agent n'a pas d'autre choix qu'accepter une offre qui lui a été faite, comme dans le cas des enchères simples sans prix de réserve. Cette modélisation est donc applicable pour les protocoles itérés où les agents peuvent formuler des offres et en accepter.

Au-delà de ces trois modules, l'agent est supposé avoir accès à un certain nombre d'éléments : sa propre fonction d'utilité, le domaine de négociation, son historique, ainsi qu'au protocole. Le domaine de négociation se compose des attributs négociés, et comprend le minimum et le maximum de l'attribut dans le cas d'un attribut quantitatif (entier ou continu) et les différentes valeurs possibles dans le cas d'un attribut catégoriel. Le protocole précise quant à lui les actions que l'agent peut faire étant donné l'état de la négociation. La Figure 3.1 donne un aperçu du modèle BOA et de l'interaction entre ses différents composants. Cette figure est une traduction de celle donnée par [Baarslag, 2016].

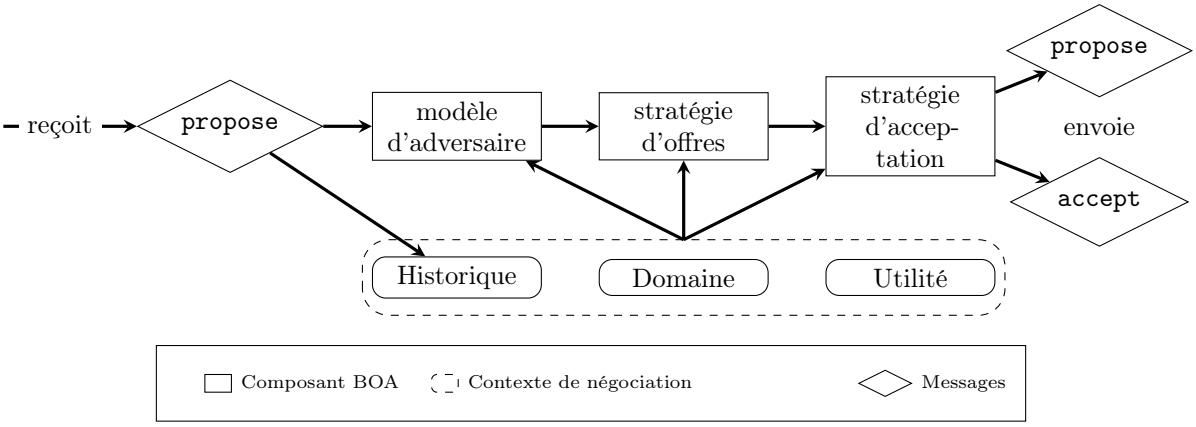


FIGURE 3.1 – Modèle BOA proposé par [Baarslag, 2016]

Ce chapitre est organisé selon le formalisme BOA. Comme le montre la Figure 3.1, les modules BOA sont dépendants les uns des autres. En particulier, la stratégie d'offres utilise la modélisation de l'adversaire et la stratégie d'acceptation utilise la proposition produite par la stratégie d'offres. Nous passons donc d'abord en revue les différents aspects de la modélisation de l'adversaire dans la littérature. Nous analysons ensuite les stratégies de proposition les plus pertinents. La Section 3.4 donne un aperçu des différentes techniques utilisées pour décider si une proposition doit être acceptée ou non. Enfin, nous concluons ce chapitre en décrivant les problématiques qui n'ont pas encore été traitées dans le cadre de la négociation automatique.

3.2 Modélisation de l'adversaire

Afin de négocier efficacement, de parvenir à un accord ou d'obtenir une utilité aussi haute que possible, il est indispensable de s'adapter à l'agent avec lequel on négocie. Cette adaptation passe par une caractérisation du partenaire de négociation, qui dans ce cadre est appelé adversaire. La modélisation de l'adversaire est un sujet central qui a été traité avec beaucoup d'attention, en premier lieu dans les programmes d'Intelligence Artificielle pour les jeux, mais aussi, plus récemment, en négociation automatique. La modélisation de l'adversaire est destinée à être utilisée par la stratégie d'offres et/ou par la stratégie d'acceptation. Les aspects de l'adversaire qui seront modélisés dépendent donc

principalement de ces stratégies.

Une classification des approches utilisées a notamment été proposée par [Baarslag *et al.*, 2015b]. Selon les particularités du protocole, plusieurs modèles peuvent être construits, afin d'inférer différents types d'informations sur l'adversaire. Elles seront utilisées afin de trouver un accord avec l'adversaire. Parmi ces informations, on retrouve :

- La stratégie d'offres : elle infère les prochaines propositions faites par l'adversaire.
- La stratégie d'acceptation : elle définit si un agent acceptera ou non une proposition.
- Le prix de réserve : dans un contexte ou le seul attribut négocié est le prix, on cherche à connaître le prix minimal ou maximal que l'agent est prêt à accepter.
- La fonction d'utilité : il s'agit d'une information plus complexe que le prix de réserve, et qui permet d'associer à chaque proposition possible une utilité inférée pour l'adversaire.
- La *deadline* : quand le protocole ne fournit pas de *deadline* publique, il est possible qu'un agent ait lui-même un délai (en temps ou en nombre de tours) au-delà duquel il annulera la négociation.

Les méthodes utilisées pour ces différentes modélisations sont différentes mais reposent toutes sur un socle commun. Pour inférer une information sur l'adversaire, l'agent doit dans tous les cas apprendre à partir des informations qu'il a : les propositions qu'il a reçues. Dans le cas où les agents négocient régulièrement les uns avec les autres, l'agent peut aussi garder l'historique des négociations avec cet agent en faisant la supposition que ce dernier ne change pas d'une session à l'autre. Les approches utilisées sont donc toutes des approches d'apprentissage automatique (*machine learning*). La suite de cette section présente les techniques utilisées par les agents pour inférer les paramètres de l'adversaire.

3.2.1 Stratégie d'offres

En fonction des suppositions faites sur l'adversaire, il est possible d'envisager de très nombreux types de méthodes d'apprentissage de la stratégie de l'adversaire. Ce modèle peut ensuite être utilisé afin d'adapter sa propre stratégie et d'obtenir le meilleur prix possible, ou encore, dans le cas de la stratégie d'acceptation, de déterminer quelle est la probabilité d'obtenir une meilleure proposition par la suite. Comme cela est expliqué dans la Section 3.3, la distinction la plus fondamentale entre les stratégies de propositions est la distinction entre stratégies fixes (aussi appelées tactiques) et les stratégies adaptatives. Déterminer la stratégie d'offres revêt une grande importance pour déterminer la stratégie à arborer, qu'il s'agisse de trouver un accord optimal pour l'agent lui-même ou pour les deux partenaires à la fois.

3.2.1.1 Les stratégies fixes

L'article [Faratin *et al.*, 1998] a eu un grand retentissement dans le domaine de la négociation automatique. Les stratégies qui y sont décrites ont été largement réutilisées, composées et modifiées par les agents de négociation automatique. La très large utilisation de ces stratégies a poussé de nombreux chercheurs à les analyser et les modéliser, afin de trouver une stratégie optimale dans le cas où l'adversaire les utilise.

Les stratégies fixes sont essentiellement des fonctions qui dépendent du temps écoulé ou du nombre de tours et de la *deadline*. Elles sont également paramétrées par plusieurs réels . La Section 3.3.1 détaille ces fonctions, qui sont issues de [Faratin *et al.*, 1998]. Si la *deadline* est disponible publiquement, l'agent a alors deux éléments à déterminer :

- la forme de la fonction d'abord. [Faratin *et al.*, 1998] proposent deux classes de fonctions, une polynomiale et une exponentielle, ainsi que plusieurs variantes prenant en compte la rareté de la ressource négociée ou du comportement de l'adversaire,
- les paramètres de cette fonction.

La Figure 3.2 représente le principe de cette technique. Dans notre exemple, nous supposons que la fonction dépend du nombre de tours écoulés, qu'il y a une unique forme de fonction possible, et que cette fonction a pour unique paramètre α. On note $f_\alpha : \mathbb{N} \to \mathbb{R}$ la fonction paramétrée associant une proposition à un tour de la négociation. Pour un tour k, on appelle p_k les propositions faites par l'adversaire. Pour chaque valeur α possible, l'agent calcule $\hat{p}_k^\alpha = f_\alpha(k)$. Il utilise ensuite une fonction de coût bien choisie C, lui permettant de calculer la distance entre la série (p_k) et la série $(\hat{p}_k^\alpha)$. Pour un historique donné, ce coût ne dépend que des propositions générée par f_α. Dans notre exemple puisqu'il n'y a qu'une forme de fonction, la fonction de coût ne dépend que du paramètre α. Le couple formé de fonction/paramètres retenu est celui qui minimise la fonction de coût.

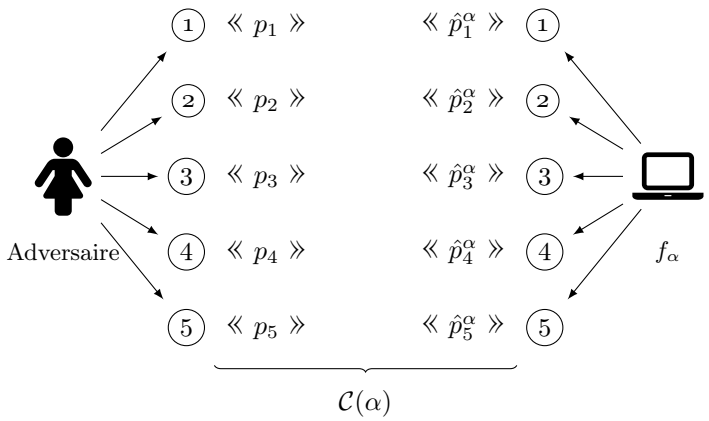


FIGURE 3.2 – Principe de la régression non linéaire

Plusieurs auteurs ont adopté cette approche. [Hou, 2004, Williams *et al.*, 2012] l'ont notamment adapté en faisant la supposition que la fonction de stratégie de l'adversaire appartient à l'une des trois familles fixes décrites par [Faratin *et al.*, 1998], les auteurs minimisent le coût pour chacune des possibilités et choisissent celle qui renvoie le coût le plus faible. La supposition que l'adversaire utilise ce genre de tactiques peut paraître simpliste, mais plusieurs articles ont montré que, si elle ne colle pas parfaitement à la réalité, elle en est en fait assez proche pour constituer une approximation suffisante. C'est le cas de [Ren et Zhang, 2007] qui utilisent uniquement une modélisation dépendant du temps écoulé, et réussissent à la généraliser à un certain nombre d'autres fonctions. Enfin [Yu *et al.*, 2013] utilisent ce type d'apprentissage dans un contexte où la stratégie des agents dépend uniquement du temps. L'agent proposé adapte alors ses propres paramètres en fonction de ceux de l'adversaire afin d'obtenir une utilité plus importante.

La régression n'est pas la seule méthode utilisable lorsqu'on fait une supposition sur les fonctions de stratégies possibles pour l'adversaire. En particulier, [Mudgal et Vassileva, 2000] caractérisent les agents selon plusieurs critères : selon qu'ils soient prêts ou non à prendre des risques, selon qu'il soit urgent ou non pour eux de conclure la négociation et selon l'importance qu'il accordent au prix dans la négociation. Ces différents paramètres caractérisent une fonction de stratégie. L'influence de ces critères est organisée sous la forme d'un réseau bayésien qui est mis à jour au fur et à mesure de la négociation.

Il est enfin possible d'utiliser des réseaux de neurones dans cette configuration. Bien que ceux-ci soient en général utilisés sans faire de supposition sur la forme de la fonction adverse (voir la Section 3.2.1.3 pour un aperçu de ces techniques), [Rau *et al.*, 2006] utilisent un réseau de neurones et proposent deux approches : une prend en entrée les dernières propositions faites par l'adversaire et l'autre un certain nombre de métriques et de ratios calculés sur ces valeurs. À partir de ces données, l'agent effectue un apprentissage sur un réseau de neurones et déduit le type de tactique utilisé et ses paramètres.

3.2.1.2 Méthodes spécifiques aux séries temporelles

Les agents les plus compétitifs ne reposent cependant pas sur des stratégies basées uniquement sur le temps passé ou une imitation stricte de l'adversaire. L'ANAC a notamment vu l'essor d'agents plus compétitifs et donc nécessairement plus complexes à modéliser. Malgré cette complexité, d'autres méthodes se sont montrées efficaces pour modéliser les adversaires, y compris si on considère qu'ils sont adaptatifs, et qu'ils changeront donc leur stratégie en fonction de leur propre adversaire. Un des principaux points commun de ces approches est qu'elles ne font aucune supposition sur la variable dont dépendent les propositions de l'adversaire (pas nécessairement du temps, ni de la ressource en particulier). Elles créent donc un lien direct entre une proposition et celles qui la suivent et la précèdent. Ce type de problématiques se retrouve dans de nombreux domaines, tant en physique qu'en finance, aussi un ensemble de techniques a t-il vu le jour pour y répondre, dont l'analyse de séries temporelles (voir [Lütkepohl, 2005] pour une introduction à ces méthodes).

Une des familles de méthodes les plus répandues pour ces problématiques est l'autorégression. Cette famille de méthodes (on trouve notamment AR, ARMA, ARIMA) considère que chaque valeur d'une série temporelle est une moyenne pondérée des éléments qui la précèdent. L'enjeu de l'apprentissage est de trouver quels éléments sont les plus pertinents et de leur attribuer le poids idéal en utilisant les données déjà acquises. L'apprentissage se fait donc sur la série elle-même. Le processus est identique à une régression, mais les paramètres de la régression sont les coefficients attribués aux éléments précédents de la série. [Chen et Weiss, 2012] utilisent cette technique pour modéliser la stratégie de l'adversaire. Les propositions sont considérées comme un signal et décomposées en utilisant l'*Empirical Mode Decomposition* [Huang *et al.*, 1998], une technique de décomposition des séries temporelles en composantes qui en facilite l'analyse. Une fois la série décomposée, c'est sur ces composantes que l'autorégression (*AutoRegressive Moving Average*) est utilisée. Cette technique donne de bons résultats face aux vainqueurs de l'ANAC 2010 et de l'ANAC 2011.

Il est cependant possible de voir les modèles d'autorégression comme un cas particulier d'un autre type de modèle d'analyse de séries temporelles : la régression de processus gaussiens (voir notamment [Rasmussen et Williams, 2006a] pour l'application à l'apprentissage). Ces modèles voient les séries temporelles comme des processus gaussiens. On peut les visualiser comme des séries temporelles dont les individus seraient non pas des valeurs, mais des gaussiennes. Il est alors possible, pour chaque valeur, de produire une prédiction sous la forme d'une gaussienne qui dépend des valeurs précédentes. Ces valeurs précédentes sont quant à elles vues comme des points, c'est à dire des gaussiennes dont la densité de probabilité est toute entière concentrée dans la valeur connue. La Figure 3.3 donne l'illustration d'une régression de processus gaussien. La figure représente des propositions faites par un agent dans un contexte mono attribut. la valeur pour le second tour de l'adversaire a volontairement été retiré afin de pouvoir observer l'effet de la présence des observations sur le processus gaussien. Comme on peut le voir, pour chaque valeur de x (ici le tour), on a une gaussienne. Les valeurs observées contraignent le processus gaussien aux tours correspondant. La régression de processus gaussiens repose sur l'utilisation d'une fonction de covariance (ou *kernel*) entre les valeurs considérées, et plusieurs fonctions

sont proposées par [Rasmussen et Williams, 2006a]. [Williams *et al.*, 2011, Williams *et al.*, 2013] proposent l'utilisation du kernel de Matérn, et montrent que cette technique est efficace face aux agents de l'ANAC 2010. [Chen et Weiss, 2013] utilisent une variante moins coûteuse en terme de calculs, qui se base sur un nombre réduit de valeurs en entrée afin de réduire le temps de calcul matriciel de la régression et qui se montre efficace face aux agents de l'ANAC 2012. Outre son efficacité, cette méthode a une particularité : elle permet, en plus d'une valeur, de prédire son degré d'incertitude. En effet, en plus de la moyenne de la gaussienne qui peut être vue comme la valeur prédite pour ARMA, la gaussienne a aussi un écart-type, qui peut être interprété comme un niveau de confiance que l'agent peut avoir dans la mesure prédite.

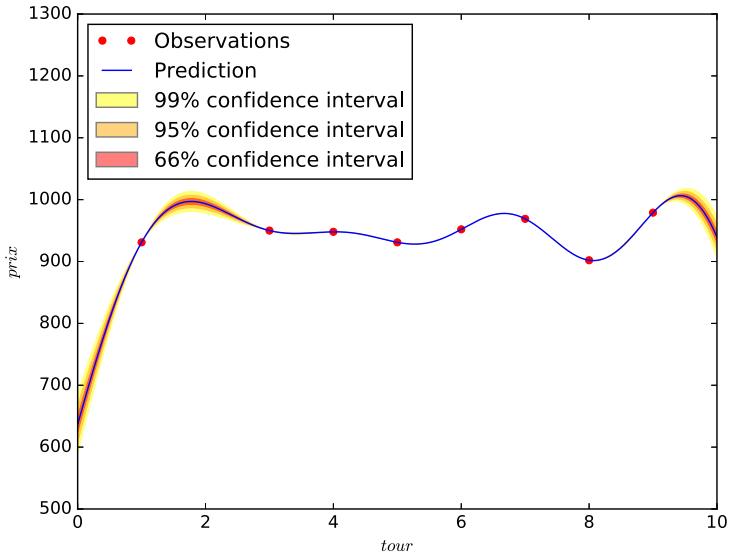


FIGURE 3.3 – Exemple de processus gaussien

3.2.1.3 Réseaux de neurones

Les réseaux de neurones ont eu beaucoup de succès récemment par exemple dans des domaines aussi divers que l'analyse d'images (voir par exemple [Egmont-Petersen *et al.*, 2002] pour plus d'informations) ou que l'IA pour les jeux (AlphaGo [Silver *et al.*, 2016] s'appuient notamment sur des réseaux de neurones profonds). La modélisation d'adversaire n'échappe pas à la règle et plusieurs travaux ont montré leur efficacité dans ce domaine. [Oprea, 2003] utilise un réseau de neurones avec une couche cachée, et dont le nombre d'unités est appris en même temps que les coefficients de chaque neurone artificiel. Cette approche est représentée en Figure 3.4. Le réseau prend uniquement les trois dernières propositions reçues pour prédire la suivante, et est applicable dans le cas mono-attribut. [Carbonneau *et al.*, 2008] utilisent une approche similaire dans le cas d'une négociation multi-attributs (l'article utilise un domaine à 4 attributs). L'entrée du réseau est composé des 39 dernières propositions de l'adversaire et l'unique couche cachée est composée de 10 unités. Le modèle est utilisé pour négocier avec des acteurs humains. Des modèles plus complexes, notamment plus profonds, ont également été proposés. [Masvoula, 2013] propose ainsi d'utiliser un réseau à plusieurs couches, dont le nombre de couches et le nombre d'offres utilisées pour modéliser l'adversaire sont déterminées par un algorithme

génétique. Enfin, il est possible d'utiliser un réseau de neurones pour déterminer quelle tactique et quels paramètres sont utilisés par l'adversaire.

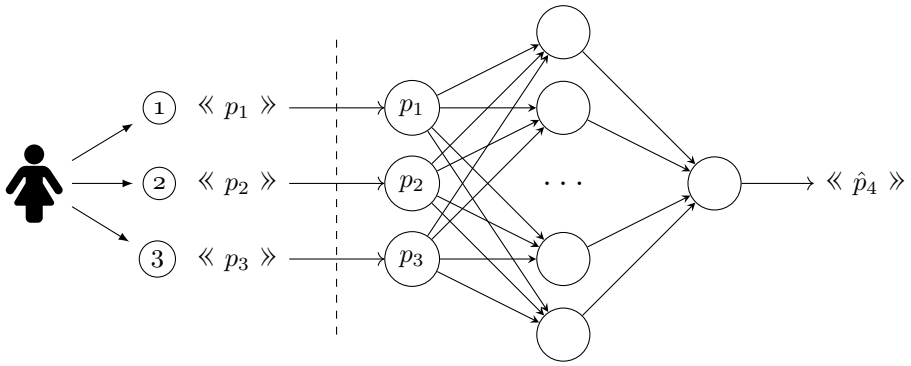


FIGURE 3.4 – Modélisation de stratégie par réseau de neurones tel que décrit par [Oprea, 2003]

3.2.2 Le prix de réserve

Le prix de réserve d'un agent par rapport à une négociation est le prix maximal auquel il est prêt à acheter l'objet s'il est acheteur, ou le prix minimal auquel il est prêt à le vendre s'il est vendeur.

L'apprentissage du prix de réserve peut se réaliser lorsque la négociation s'effectue uniquement sur un prix (négociation mono-attribut). Dans ce cas, le prix de réserve est une information extrêmement précieuse : il permet de savoir quelles sont les limites de l'adversaire et donc de s'en approcher. En s'approchant de ces limites, il est possible de s'assurer de remporter la négociation : en négociant près de ce prix, on obtient le meilleur prix que l'adversaire est prêt à concéder.

Là encore, si on suppose que l'adversaire utilise une tactique prédéfinie, telle que définie dans la Section 3.2.1.1, alors, le prix de réserve est en fait un des paramètres de la stratégie (voir la Section 3.3.1 pour les détails). Les approches qui sont décrites dans la Section 3.2.1.1 permettent alors d'estimer la stratégie de l'adversaire, mais aussi son prix de réserve.

Pour apprendre le prix de réserve, [Zeng et Sycara, 1998] proposent une approche bayésienne. Un ensemble d'hypothèses est généré *a priori* sur le prix de réserve. Les auteurs font la supposition que l'agent ne fera pas de proposition au-delà de son prix de réserve. Dans ce cas, chaque proposition faite par l'adversaire vient mettre à jour les probabilités de chaque hypothèse sur le prix de réserve. Cette approche est utilisée par leur agent Bazaar.

Il est particulièrement notable que si les agents font la supposition que la tactique de l'adversaire est une fonction d'une forme connue, l'apprentissage bayésien permet de trouver son prix de réserve d'autant plus efficacement. En effet, ces fonctions convergent vers le prix de réserve. C'est l'approche qu'utilisent par exemple [Sim *et al.*, 2009] : leur agent BLGAN utilise lui aussi une approche bayésienne : les probabilités sont mises à jour en faisant la supposition que les agents suivent une stratégie ayant une forme connue.

3.2.3 La fonction d'utilité

La fonction d'utilité est la fonction qui évalue la satisfaction d'un agent pour toute issue d'une négociation. On considère traditionnellement que l'utilité nulle représente le critère d'indifférence : l'agent ne gagne rien ni ne perd rien. Si l'agent est rationnel, le prix de réserve peut alors être considéré comme le prix garantissant à l'agent l'utilité dans le cas où la négociation échoue : un agent refusera toute proposition lui procurant une utilité inférieure. Au contraire, il préférera obtenir cette utilité plutôt que faire échouer la négociation. Dans les faits, on considère souvent que l'utilité de réserve est 0.

Apprendre la fonction d'utilité de l'adversaire a plusieurs avantages. Cela permet d'une part de vérifier si une proposition est acceptable par l'adversaire ou non, et ainsi d'augmenter ses chances d'arriver à un accord. La connaissance de cette fonction peut également être utilisée comme le prix de réserve pour déterminer quelle est la meilleure offre que l'adversaire est prêt à accepter.

Dans un domaine de négociation multi-attributs, la fonction d'utilité est très complexe à inférer. En l'absence d'historiques sur les négociations précédentes, il est impossible d'apprendre cette fonction sans faire de présuppositions. La première est que la fonction d'utilité est décomposable en somme pondérée de fonctions d'utilité pour chaque attribut. Ainsi, pour un domaine constitué de n attributs, la fonction d'utilité u d'un agent peut être décomposée ainsi :

$$u = \sum_{k \leq n} w_k \cdot u_k \tag{3.1}$$

où les w_k sont des poids se sommant à 1. Toutes les approches que nous décrivons dans cette section font des suppositions sur la stratégie de l'adversaire. En effet, si on ne fait pas cette supposition, il est impossible de faire une association entre les informations disponibles (les propositions faites par l'adversaire) et l'utilité. En général, les différentes approches considèrent que l'adversaire fait des concessions d'une proposition à la suivante.

Il existe cependant un certain nombre de travaux dont le but a été d'inférer, dans un cas où la fonction d'utilité n'est pas ainsi décomposable, comment les valeurs des différents attributs peuvent interagir et influencer la fonction d'utilité. [Robu et La Poutré, 2005] proposent un modèle où les agents négocient un *cluster* de bien (de la même façon que dans les enchères combinatoires, *cf.* Section 2.2.1.2). Le nombre de propositions possibles croît de manière exponentielle en fonction du nombre d'items à négocier. Le modèle de l'adversaire consiste en un graphe complet dont les nœuds sont les items et dont les arêtes sont étiquetées par le surplus d'utilité causé par la présence de ces deux items dans un même *cluster*. Les poids des arêtes sont mis à jour à chaque réception d'une proposition : tous les couples formés par deux éléments quelconques du *cluster* de la propositions sont augmentés tandis que les poids de toutes les autres arêtes sont diminués. [Kakimoto et Fujita, 2016] prennent eux aussi en compte les interactions entre attributs deux à deux, mais ils utilisent une méthode à base de fréquences. Le nombre d'occurrences de chaque couple de valeurs est observé et une base de données est maintenue à jour. L'utilité apportée par un couple à l'agent est fonction de sa fréquence. Notons que ces méthodes permettent de représenter le surplus de biens induit par la présence de deux biens dans le même *cluster*, mais elles ne différencient pas, par exemple, si un trio de biens est intéressant ou si ce sont plutôt les duos formés par chaque paire d'items du trio qui intéressent l'adversaire.

Il y a donc plusieurs éléments à déduire : d'abord, le poids de chaque attribut. Ensuite, la fonction d'utilité liée à chacun de ces attributs. En ce qui concerne les poids, [Faratin *et al.*, 2002] a montré que la connaissance de l'ordre des attributs pour l'adversaire permet déjà d'améliorer l'efficacité des propositions faites. Il est alors possible pour une méthode qui ne traite pas ce problème, et d'utiliser

une formule exprimant les poids de manière arbitraire en fonction du rang de chaque attribut, en vérifiant que ces poids se somment bien à 1.

Inférer l'ordre des attributs peut être fait de plusieurs manières. [Niemann et Lang, 2009] utilisent une technique basée sur de l'apprentissage bayésien. Comme dans les approches utilisées pour l'apprentissage du prix de réserve, un ensemble d'hypothèses est généré, et mis à jour au fur et à mesure de la négociation. À chaque fois qu'une proposition est reçue, la probabilité de chaque hypothèse est mise à jour. Les auteurs appliquent cette méthode dans un contexte où les agents ont beaucoup d'information sur les fonctions d'utilité associées à chaque attribut (leur sens de variation et les valeurs acceptables). On peut cependant se demander si cette méthode est réaliste : de telles informations sont rarement connues dans les cas réels, et elles sont nécessaires pour faire les calculs dans le cadre de cette méthode. La supposition faite par les auteurs est que le taux de concession est calculable par attribut et inversement proportionnel au poids associé à l'attribut.

Certaines méthodes se proposent de déduire à la fois le poids de chaque attribut et la fonction correspondante. [Lin *et al.*, 2006] proposent d'associer la fonction d'utilité d'un agent à une fonction appelée nombre de Luce (d'après [Luce, 2005]). Cette fonction croît avec la probabilité que l'adversaire accepte l'offre qui lui est faite. Elle permet donc de faire de l'apprentissage bayésien dans un contexte favorable : les hypothèses sont restreintes à ces nombres, finalement peu nombreux. [Hindriks et Tykhonov, 2008] proposent eux aussi de s'appuyer sur l'apprentissage bayésien, mais la seule supposition qu'ils font sur la fonction d'utilité est qu'elle est décroissante en fonction du temps, et que sa décroissance se fait à un rythme fixe. Chaque hypothèse consiste à supposer que la fonction d'utilité se décompose en une somme pondérée de fonctions pour chaque attribut. Ces fonctions sont de simples fonctions triangulaires dont le maximum atteint 1. Le poids de chaque attribut est déduit de l'ordre d'importance qui lui est assigné selon une formule arbitraire, en utilisant le résultat de [Faratin *et al.*, 2002] décrit dans le paragraphe précédent. Les probabilités sont mises à jour, comme chez [Lin *et al.*, 2006], à chaque proposition reçue. La fonction d'utilité prédite est la somme des différentes hypothèses, pondérées par la probabilité de chacune. Les auteurs montrent que le fait de sommer ainsi plusieurs fonctions triangulaires peut générer des fonctions beaucoup plus complexes. Le processus est représenté sur la Figure 3.5.

La réception d'une proposition par l'adversaire nous apprend en fait plus que le simple fait qu'une offre est intéressante pour lui. [Aydoğan et Yolum, 2012] considèrent que l'offre qu'il fait est plus intéressante pour lui que les offres qu'il a reçues (dans le cas inverse, il aurait dû accepter une offre précédente). En comparant ce qui est envoyé et ce qui est reçu, les auteurs utilisent l'apprentissage inductif pour tirer des conclusions sur les préférences de l'agent pour chaque domaine. Cette approche est intéressante, mais elle ne permet ni de déduire un ordre sur les propositions ni de savoir ce qui est acceptable ou non pour l'agent. Elle a pour particularité de ne pas faire la supposition que l'adversaire fait des concessions. Le cadre est plutôt d'explorer le domaine de négociation afin de trouver une négociation qui satisfasse les deux agents.

Enfin, certaines heuristiques se sont montrées particulièrement efficaces pour prédire les préférences d'un agent. Comme l'approche décrite précédemment, celle développée par [van Galen Last, 2012] observe la fréquence de chaque valeur pour chaque attribut. Cette observation permet à l'auteur de faire deux déductions. D'abord les valeurs ayant la fréquence la plus élevée dans les propositions adverses sont celles qu'il préfère. Ensuite, les attributs dont la valeur change le plus sont les moins importants pour l'adversaire : ils sont pour lui un facteur d'ajustement, là où faire une concession sur un autre attribut lui est plus coûteux, ce qu'il fera donc plus rarement. L'agent ABiNeS, proposé par [Hao et Leung, 2012] fait quant à lui la supposition supplémentaire que l'adversaire fait des concessions et donc que les propositions antérieures ont une plus grande valeur pour l'agent. Ces approches

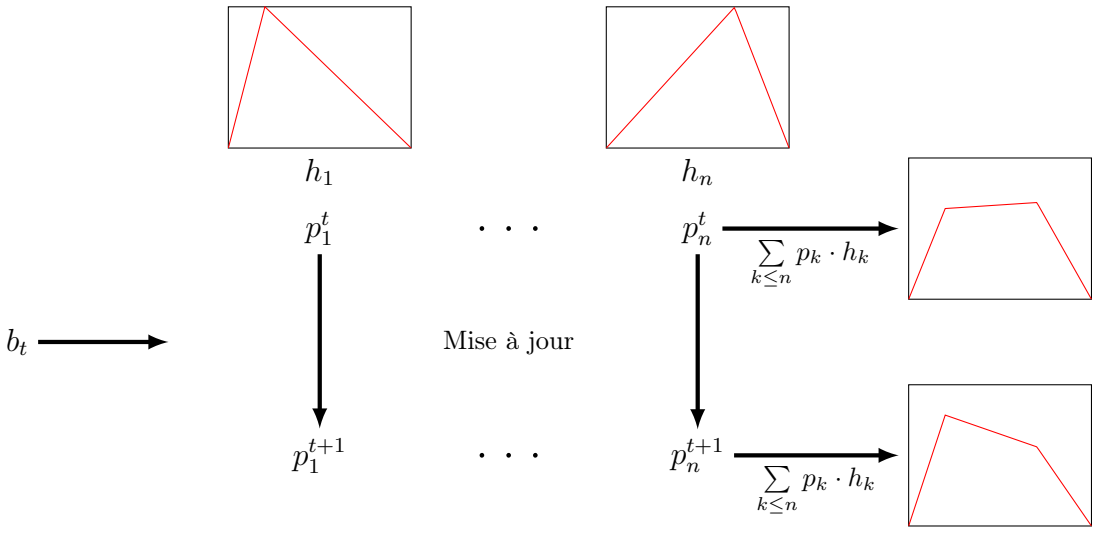


FIGURE 3.5 – Représentation de la modélisation de stratégie par un apprentissage Bayésien, décrit par [Hindriks et Tykhonov, 2008]

ne sont cependant valables que dans le cas où le domaine de négociation est discret, voire entier. Il n'est en aucun cas applicable à des domaines continus. Cette technique a été largement utilisée, notamment par BraveCat [Zafari et Nassiri-Mofakham, 2016].

Pour les domaines numériques, il est nécessaire de pouvoir mesurer la distance entre deux valeurs proposées. Cette question est complexe. Même dans le cas où un domaine a une valeur maximale et une valeur minimale, la distance doit être établie non en fonction de la distance euclidienne entre des nombres, mais selon la représentation que s'en fait l'adversaire. En effet, même sur un domaine allant de 0 à 1000, les valeurs acceptables pour l'adversaires peuvent tout à fait être comprises entre 5 et 10, avec un maximum à 7. Dans ce cas, la distance de 5 à 7 est très supérieure à celle de 12 à 1000, puisque cette distance est de son point de vue composée de valeurs inacceptables. [Buffett *et al.*, 2006] utilise cette approche, dans un contexte où une fonction de similarité est connue, afin de savoir si une proposition constitue ou non une concession pour l'adversaire.

3.2.4 Stratégie d'acceptation

La stratégie d'acceptation est la stratégie qui décide si un agent devrait ou non accepter une offre qui lui a été faite. La connaissance de cette stratégie est généralement utilisée pour trouver les offres qui ont le plus de chance d'être acceptées, et donc pour réduire les risques de terminer la négociation sans accord.

Comme le montre la Section 3.4, nombre de stratégies d'acceptations sont en réalité très simples, et dépendent uniquement de la dernière proposition qui a été faite et de l'utilité de l'agent. Dans le cas où une telle supposition est faite, un agent ayant à sa disposition la fonction d'utilité de son adversaire (*cf.* Section 3.2.3) peut déduire si l'agent adverse va ou non accepter une proposition arbitraire, à condition que sa supposition se trouve être juste et que son modèle soit assez précis. Ces deux conditions sont dans les faits difficiles à remplir et il existe donc d'autres modèles permettant une modélisation plus

fine.

En dehors de ces approches, l'apprentissage de la stratégie d'acceptation se fait sous forme de probabilité, et implique la connaissance des négociations précédentes. [Oshrat *et al.*, 2009] proposent ainsi d'apprendre l'utilité de l'adversaire en s'appuyant sur [Hindriks et Tykhonov, 2008] et, en utilisant l'historique de négociations antérieures, de déterminer quelle utilité l'adversaire est prêt à accepter. [Fang *et al.*, 2008] utilisent une approche assez différente en considérant que chaque proposition faite par l'adversaire est également acceptable de son point de vue. Un réseau de neurones artificiel est entraîné sur toutes les propositions faites par l'adversaire lors de cette négociation et des précédentes et donne en sortie un booléen indiquant si la proposition est acceptée ou non. Enfin, [Saha *et al.*, 2005] approximent la fonction de probabilité d'acceptation d'un adversaire pour une négociation mono-attribut en l'interpolant avec des polynômes de Tchebychev. Un exemple d'interpolation par des polynômes de Tchébychev est représenté en Figure 3.6. Comme le montrent la figure et [Mason et Handscomb, 2002], cette approximation est assez efficace dès lors que le nombre de polynômes est suffisant. Les données utilisées pour l'interpolation sont les propositions acceptées lors de précédentes négociations.

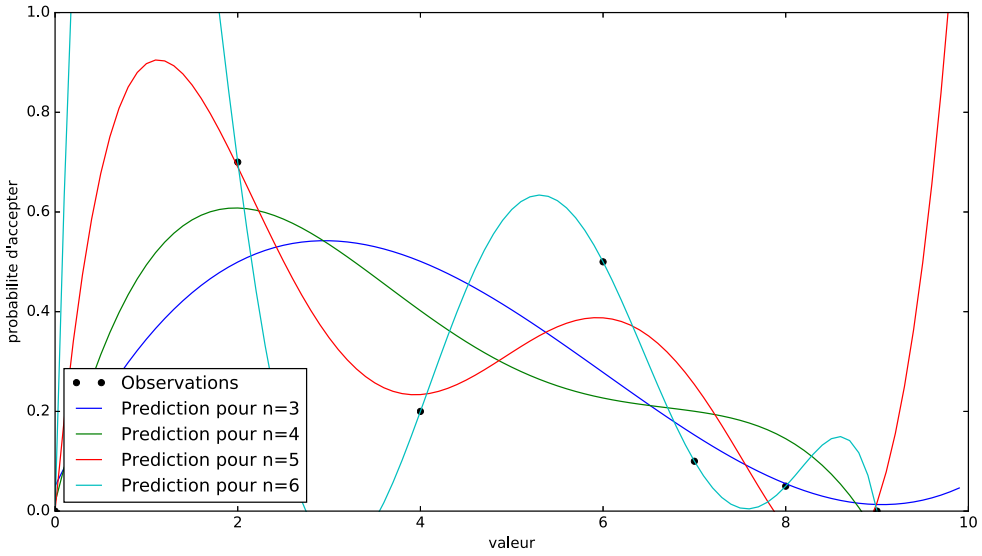


FIGURE 3.6 – Interpolation d'une fonction par n polynomes de Tchebychev

3.2.5 Deadline

Dans un contexte où aucune limite ni en temps ni en nombre de tours n'est spécifié par le contexte de négociation (typiquement, dans un protocole avec un nombre de tours potentiellement infini pour reprendre la classification donnée en Section 1.2.2), il est toujours possible pour un agent de fixer sa propre limite, un délai ou un nombre de tours qu'il ne souhaite pas dépasser pour acheter ou vendre un bien. Il est possible d'apprendre cette borne. Cela peut permettre d'adapter sa stratégie, et de faire davantage de concessions lorsque s'approche cette limite, afin d'arriver à un accord. L'apprentissage de la *deadline* sans aucune connaissance préalable sur l'adversaire n'est pas possible sur une unique négociation, et supposer, comme pour la stratégie d'acceptation en Section 3.2.4, qu'elle ne changera

pas d'une négociation sur l'autre est coûteux. Le temps que se laisse un agent pour négocier dépend de l'urgence qu'il a à arriver à un accord, et cela dépend du bien négocié, de l'agenda de l'agent, de ses besoins... En faisant des suppositions sur la forme de la fonction utilisée, les techniques présentées pour apprendre la stratégie de l'adversaire en Section 3.2.1.1 restent applicables et voient la *deadline* comme un paramètre, appris lui aussi.

3.2.6 Discussion

La modélisation de l'adversaire peut être utilisée à différentes fins : il est par exemple possible d'apprendre le profil de préférence de l'adversaire afin d'augmenter les chances de trouver un accord. Il peut tout aussi bien s'agir de déterminer le prix de réserve de l'adversaire afin de trouver quel est le meilleur prix que l'on peut obtenir de la négociation, aux dépens de l'adversaire.

Dans tous les cas, qu'il s'agisse de réseaux de neurones, de régression non linéaire, d'autorégression ou d'apprentissage bayésien, la mesure sera d'autant plus fiable que l'agent a des données (voir [Halevy *et al.*, 2009] sur l'efficacité des données sur les algorithmes d'apprentissage). Cela peut avoir une incidence sur le comportement des agents. En effet, dans le cas où un agent souhaite avoir un modèle fiable d'un adversaire, il a une incitation à obtenir des informations de cet adversaire. Si cette incitation est forte, elle peut même pousser l'agent à agir de manière nuisible. Ce sont ces comportements que nous décrivons dans la Chapitre 4.

Cela ne signifie pas que l'apprentissage soit mauvais en soi. Dans des domaines de négociation complexes, connaître l'utilité de l'adversaire peut permettre à un agent de faire des propositions efficaces, au sens où elles profitent à la fois à son adversaire et à lui-même, en s'approchant de la frontière de Pareto. Le problème tient davantage aux déviations que cela peut entraîner vis à vis d'un agent qui souhaite garder certaines informations secrètes, comme typiquement son prix de réserve.

3.3 Stratégies d'offres

Une fois l'adversaire modélisé, l'information retirée du modèle est utilisée pour produire une proposition. Dans cette section, nous donnons un aperçu des approches utilisées en négociation automatique à cette fin. De nombreux travaux ont été menés pour concevoir des stratégies efficaces. Certaines sont totalement indépendantes d'un quelconque modèle de l'adversaire. Certains protocoles imposent aux agents négociant de faire des concessions à chaque tour. Toutefois, même lorsque cela n'est pas imposé par le protocole, il est courant que les agents fassent cette supposition eux-même. Dans le cas où cette règle a du sens, de nombreux agents l'utilisent comme base pour construire leur stratégie. Cependant, notamment dans un contexte où le domaine de négociation est complexe, faire une concession peut nuire à la fois à l'agent et à son adversaire. Dans ce cas, il est nécessaire de construire des agents qui utilisent d'autres techniques. Si les stratégies utilisées aujourd'hui sont issues de techniques complexes, d'autres sont beaucoup plus simples, et ne dépendent que du nombre de tours écoulés.

3.3.1 Stratégies fixes

Un des travaux fondamentaux en négociation automatique a été réalisé par [Faratin *et al.*, 1998]. Dans cet article, les auteurs font la supposition que la stratégie de l'adversaire peut dépendre de trois

éléments :

- le temps écoulé (par rapport au temps ou au tour limite),
- la ressource négociée,
- le comportement de l'adversaire.

Les stratégies présentées dans cet article ont été largement réutilisées au moins dans leur principe, si ce n'est par leur implémentation. Nous les décrivons dans la suite de cette section.

3.3.1.1 Principes

En ce qui concerne la stratégie dépendant du temps écoulé, les auteurs proposent deux familles de fonctions : les fonctions polynomiales et les fonctions exponentielles. Les fonctions dépendent toutes du temps écoulé (ou du nombre de tours passés t), d'un paramètre β, et de la *deadline* t_{max}. Supposons qu'on négocie le prix d'un objet, la stratégie s'exprime par la fonction $b(t)$, qui définit la proposition faite par l'agent au temps t :

$$b(t) = \begin{cases} min + \alpha(t) \cdot (max - min) \text{ pour un acheteur} \\ min + (1 - \alpha(t)) \cdot (max - min) \text{ pour un vendeur} \end{cases} \tag{3.2}$$

où min et max sont les limites du domaine de négociation (le prix minimal et le prix maximal dans le cas où l'attribut est le prix). C'est la fonction α qui va différencier les deux types de tactiques. Ainsi, la tactique polynomiale a pour fonction α :

$$\alpha_{poly}(t) = \kappa + (1 - \kappa) \cdot \left(\frac{t}{t_{max}}\right)^{\frac{1}{\beta}} \tag{3.3}$$

La tactique exponentielle peut quant à elle être exprimée comme :

$$\alpha_{exp}(t) = \exp\left(\left(1 - \frac{t}{t_{max}}\right)\beta \ln(\kappa)\right) \tag{3.4}$$

où κ représente le prix de réserve de l'agent, exprimé en fonction des bornes du domaine de négociation : $RP = min + (max - min) \cdot \kappa$.

Pour un agent ayant une *deadline* (privée ou publique) et un prix de réserve définis, c'est le paramètre β qui décide de la forme de la fonction. La Figure 3.7 montre l'influence de β sur les stratégies polynomiales. Si $\beta > 1$, l'agent concède beaucoup dès le début de la négociation, et la stratégie est dite « *conceder* ». Si $\beta < 1$, l'agent concède très peu au début de la négociation. Mais ce faisant, l'agent prend le que l'adversaire abandonne la négociation en voyant que l'agent ne fait pas de concession. De même, l'agent risque que l'adversaire refuse sa dernière offre (parce qu'elle n'est pas acceptable pour lui) et que la négociation soit interrompue par la *deadline*. Enfin, le cas $\beta = 1$, l'agent concède de manière linéaire.

Les auteurs proposent également d'adapter la stratégie en fonction dans le contexte d'un marché de ressources où les agents ont plusieurs alternatives pour acheter ou vendre leur bien. Dans ce cas, la fonction peut aussi dépendre de la disponibilité de la ressource, c'est à dire le nombre d'unités de ressources en vente au tour considéré. À cette fin, ils font deux propositions : établir une fonction particulière qui dépend de la disponibilité de la ressource, ou adapter la *deadline* de l'agent dans les fonctions dépendantes du temps. La tactique dépendant de la disponibilité de la ressources est identique à celle décrite en Équation (3.2), mais où la fonction α est remplacée par :

$$\alpha(t) = \kappa + (1 - \kappa) \exp\left(-r(t)\right) \tag{3.5}$$

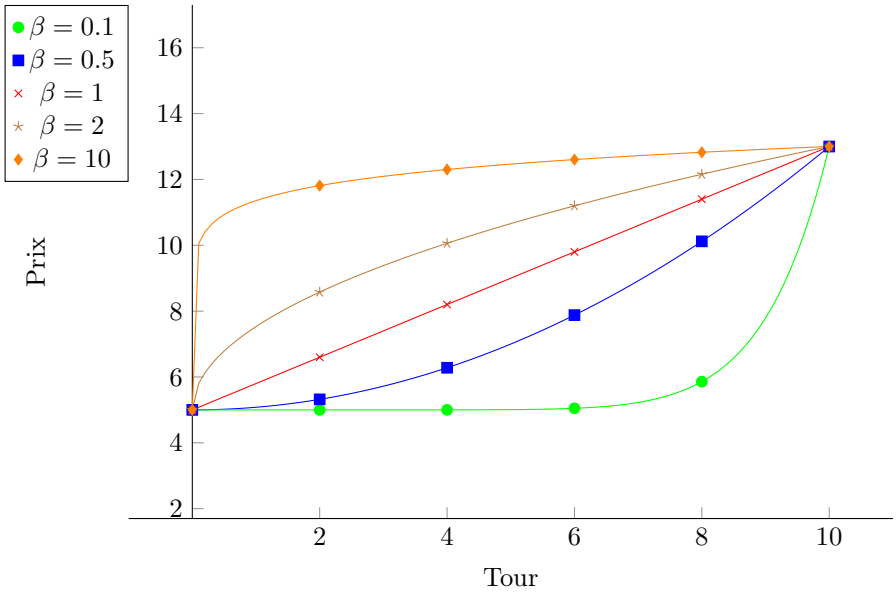


FIGURE 3.7 – Influence de β sur la stratégie polynomiale pour un acheteur

avec les mêmes notations que précédemment, et où la fonction $r(t)$ représente la disponibilité de la ressource (le nombre d'unités en vente) au tour t. Cela suppose en particulier que l'agent soit capable d'estimer cette quantité.

Bien que [Faratin *et al.*, 1998] se concentrent sur la négociation bilatérale, ils proposent également une stratégie pour le contexte multilatéral. Cette stratégie ne se concentre sur aucun protocole précis, mais sur le contexte multilatéral en général. Dans ce contexte, il est possible de mesurer la disponibilité d'un bien du point de vue d'un agent en mesurant le nombre d'adversaires négociant de manière simultanée avec lui. Il est possible de reprendre les Équations (3.3) et (3.4), mais en remplaçant le t_{max} par une fonction du nombre d'adversaires négociant avec l'agent. On appelle N le nombre d'adversaires avec lequel l'agent négocie en parallèle, $|h_i|$ la taille de l'historique de négociation avec l'adversaire i. On définit en outre une valeur de référence t^0_{max} correspondant à la *deadline* initiale dans le cas où l'agent ne négocie qu'avec un adversaire. La fonction t_{max} s'écrit

$$t_{max}(t) = t^0_{max} \cdot \frac{N^2}{\sum\limits_{i \leq N} |h_i|} \tag{3.6}$$

Il est à noter que plus la *deadline* augmente, moins l'agent concède (en supposant que les autres paramètres sont fixés). Cette tactique permet à l'agent d'allonger la négociation quand il y a beaucoup d'agents avec lesquels négocier (*i.e.* la ressource est abondante) et de la diminuer quand les historiques sont longs. l'agent concède davantage au fur et à mesure que les négociations se terminent, tant qu'il n'a pas obtenu la quantité de ressource souhaitée.

La dernière variante proposée par les auteurs dépend du comportement de l'adversaire. Il s'agit pour l'agent de l'imiter (stratégie dite *tit-for-tat*). L'agent va donc faire une concession *identique* à celle de l'adversaire, soit relativement, soit dans l'absolu, soit par rapport à la dernière proposition, soit sur une période plus longue.

La première variante de ce type de stratégies imite le comportement adverse en relatif, sur une

unique concession faite il y a δ tours. L'agent fait la proposition $b(t)$ vérifiant :

$$b(t) = \frac{b_{opp}(t-\delta-1)}{b_{opp}(t-\delta)} \cdot b(t-1) \tag{3.7}$$

où $b_{opp}(t-\delta-1)$ est la proposition faite par l'adversaire δ tours auparavant et $b(t-1)$ est la proposition faite par l'agent au tour précédent.

La seconde variante de ce type de stratégies imite le comportement adverse en absolu, sur une unique concession :

$$b(t) = \begin{cases} b(t-1) + b_{opp}(t-\delta-1) - b_{opp}(t-\delta) + R(M) \text{ pour un acheteur} \\ b(t-1) + b_{opp}(t-\delta-1) - b_{opp}(t-\delta) - R(M) \text{ pour un vendeur} \end{cases} \tag{3.8}$$

avec les mêmes notations et où R est la fonction aléatoire et M est un paramètre de l'agent décidant s'il doit exactement imiter l'adversaire ou non. La différence avec l'équation précédente est que l'agent imite l'adversaire non plus dan la proportion de concession qu'il fait mais dans la valeur de la concession.

Enfin, la dernière tactique consiste à imiter le comportement de l'adversaire moyenné sur δ tours.

$$b(t) = \frac{b_{opp}(t-\delta-1)}{b_{opp}(t)} \cdot b(t-1) \tag{3.9}$$

Cette dernière stratégie considère l'évolution la concession en proportion qui a été faite sur les δ derniers tours et l'imite. Cela permet notamment d'éviter de changer brusquement le taux de concession même lorsque l'adversaire le fait.

Notons que ces tactiques sont conçues pour faire de la négociation sur le prix. Dans le cas où le domaine est plus complexe, il reste utilisable si on est toujours capable d'attribuer une utilité à chaque proposition faite par l'adversaire. Dans le cas contraire, il est impossible d'associer une valeur à une proposition, et donc d'utiliser les fonctions b et b_{opp}.

3.3.1.2 Améliorations

Dans le cas où la négociation ne porte pas directement sur un prix, il est possible de remplacer les fonctions b et b_{opp} par les fonctions d'utilité respectivement de l'agent et de l'adversaire. Les fonctions précédemment décrites sont alors utilisées pour choisir l'utilité de la prochaine proposition faite par l'agent. Une fois cette utilité déterminée, l'agent choisit une proposition qui y correspond dans le domaine de négociation.

WhaleAgent, développé par [Sato et Ito, 2016] implémente ce genre de stratégies, basée à la fois sur le temps écoulé et sur l'imitation de l'adversaire. Si l'adversaire fait des propositions proches en terme d'utilité, l'agent utilise une stratégie qui fait très peu, voire pas de concession. Si au contraire, l'adversaire fait des propositions pour lesquelles l'utilité de l'agent augmente, celui-ci adopte une stratégie où il fait davantage de concessions. Ces deux stratégies reposent sur une connaissance de la deadline. La recherche de nouvelles propositions se fait en utilisant l'heuristique de « *hill climbing* » (*cf.* [Selman et Gomes, 2006]) d'une part et l'heuristique de recuit simulé d'autre part.

3.3.2 Stratégies adaptatives utilisant des stratégies fixes

Les tactiques ont été largement étudiées, tant et si bien que, comme cela est décrit dans la Section 3.2, un certain nombre de techniques de modélisation de l'adversaire se sont concentrées exclusivement dessus. Certaines stratégies se sont donc proposées de les utiliser comme base et d'y intégrer une composante adaptative issue d'un modèle d'adversaire (typiquement un modèle de la Section 3.2), ou encore de les modifier afin de les adapter au contexte.

3.3.2.1 Optimisation de la probabilité d'acceptation

La modification la plus élémentaire et nécessaire concerne les domaines multi-attributs. Dans ce cas, les tactiques citées ci-dessus sont utilisées pour déterminer une utilité. On recherche ensuite une proposition correspondant qui maximise sa chance d'être acceptée en optimisant soit l'utilité adverse, soit sa probabilité d'acceptation (voir Sections 3.2.3 et 3.2.3). C'est ce que fait HardHeaded développé par [van Krimpen *et al.*, 2013].

L'agent Nice Tit-for-tat, développé par [Baarslag *et al.*, 2013] est assez similaire, à cela près qu'il se repose principalement sur la stratégie d'imitation relative et non sur la stratégie dépendant du temps. Cette stratégie est modifiée afin que la concession accordée par l'agent soit bornée. Ainsi, l'agent concède nécessairement moins que l'adversaire.

AgentYk, conçu par [Kadono, 2016] propose également une stratégie issue de l'imitation de l'adversaire : elle repose d'une part sur l'utilité de l'agent, et d'autre part sur les propositions faites par l'adversaire auparavant : l'agent sanctionne les propositions qui ne font pas de concessions et sont moins avantageuses pour lui. Cependant, selon la distance à la borne, l'agent accepte que l'adversaire fasse des "erreurs", et les pardonne, acceptant de faire à nouveau des concessions si l'adversaire en fait lui aussi à nouveau. Une fois l'utilité de l'agent fixée, celui-ci choisit une proposition en fonction de son modèle d'utilité.

3.3.2.2 Modification en fonction du stade de la négociation

La pression exercée par le temps sur les agents augmente au fur et à mesure que la *deadline* s'approche. Certains agents ont donc proposé de faire varier la stratégie (fixe ou adaptée d'une stratégie fixe) en distinguant des stades de négociation.

La stratégie d'offres de BraveCat, l'agent développé par [Zafari et Nassiri-Mofakham, 2016] repose sur cette idée. BraveCat est conçu pour la négociation sur un domaine de négociation large où les fonctions d'utilité peuvent ne pas être linéaires, et où la *deadline* est connue. Sa stratégie d'offre se décompose en deux phases :

- dans la première phase, l'agent commence par générer une utilité cible selon une stratégie fixe dépendant du temps. Il modifie la valeur obtenue de manière aléatoire et ajoute une composante de *tit-for-tat*. Il obtient une valeur qui constitue l'utilité de la proposition qu'il va envoyer. Il génère ensuite un certain nombre de propositions lui procurant cette utilité et choisit celle qui maximise le modèle de l'utilité de l'adversaire,
- dans les derniers tours de la négociation, BraveCat reprend les propositions de l'adversaire qui lui ont conféré la meilleure utilité et les envoie à son adversaire.

La stratégie de DoNA, proposé par [Erez et Zuckerman, 2016] est proche : elle dépend du temps et du prix de réserve de l'agent. L'agent utilise 3 stratégies prédéterminées qui sont par la suite combinées

pour créer une stratégie composite. L'agent bascule d'une stratégie à l'autre selon la distance à la *deadline* et son prix de réserve. Les stratégies définissent la valeur de l'utilité de la proposition retenue. Afin de décider de la proposition à faire, l'agent génère 100.000 propositions et en choisit une qui correspond à l'utilité souhaitée.

3.3.2.3 Variation des paramètres

Certains agents utilisent une autre approche dérivée des stratégies fixes. Ces approches font varier le taux de concession au cours de la négociation. Cette variation est identique à celle proposée par [Faratin *et al.*, 1998] pour faire varier en fonction de la rareté de la ressource.

C'est par exemple le cas d'AgentK, développé par [Kawaguchi *et al.*, 2012, Kawaguchi *et al.*, 2013]. Son taux de concession varie en fonction de la meilleure concession attendue de l'adversaire et de la variété des offres de ce dernier. Cette stratégie est corrigée afin que l'agent ne concède pas trop quand l'adversaire négocie durement. Une correction est aussi faite dans les derniers tours afin de concéder plus finement. Le fonctionnement de IAMhaggler [Williams *et al.*, 2012] est similaire : l'agent modélise la stratégie de l'adversaire, et trouve l'utilité maximale qu'il peut attendre ainsi que le moment où il l'atteindra. Il calque ensuite le paramètre de la fonction fixe, β, de manière à atteindre ce point. S'il y a plusieurs attributs, cette fonction représente l'utilité qu'il veut atteindre, et c'est l'offre ayant cette utilité et maximisant le modèle d'utilité de l'adversaire qui est faite. [Yu *et al.*, 2013] utilisent eux aussi un modèle d'apprentissage : en fonction de la modélisation de la stratégie de l'adversaire, ils calculent le β qui maximise leur utilité.

IAMhaggler2011, conçu par [Williams *et al.*, 2013] utilise une technique similaire, mais puisque son modèle repose sur une régression de processus gaussien, il prend aussi en compte le risque pour déterminer l'objectif qu'il désire atteindre. Il fait ensuite une concession linéaire vers ce point (qui est mis à jour à chaque proposition). OMAC [Chen et Weiss, 2014] et FawkesAgent [Koeman *et al.*, 2015], reposent sur une technique d'apprentissage différente, mais qui propose également une estimation de l'écart type, et procèdent de la même manière. FawkesAgent se distingue cependant d'OMAC en concédant plus rapidement quand la *deadline* s'approche. De plus, comme les agents cités plus haut dans cette section, il est doté d'un module d'apprentissage de l'utilité de l'adversaire et recherche parmi les propositions de même utilité celle qui maximise l'utilité de l'adversaire.

Yushu [An et Lesser, 2012] adopte une stratégie basée sur une stratégie fixe dépendant du temps, mais il règle son taux de concession sur la compétitivité du domaine, un indice qu'il mesure en vérifiant la proportion des propositions de l'adversaire lui apportant une faible utilité. [Dirkzwager et Hendrikx, 2014] distinguent les agents en fonction de leur disposition à faire ou non des concessions et de l'existence d'un *discount factor*. Il met son taux à jour selon une table déterminée de manière empirique. [Morii et Ito, 2014] utilisent le même type de technique, adaptant leur prix de réserve théorique à un indice dépendant de la manière dont l'adversaire concède, mais leur fonction de négociation (dépendante du temps) ne repose pas sur les fonctions présentées dans la Section 3.3.1, mais sur une fonction sigmoïde.

AgentM, l'agent développé par [Niimi et Ito, 2016] pour les domaines larges et les utilités non linéaires repose sur une stratégie de type tit-for-tat (imitation du comportement de l'adversaire), mais plus complexe. L'agent prend en compte l'écart entre les propositions adverses, la meilleure proposition qu'il a faite, la meilleure proposition reçue et la proposition "typique de l'adversaire". En outre, la stratégie prend aussi en compte le tour de la négociation et le borne afin de produire une nouvelle proposition. Cette stratégie, bien qu'empirique, s'est montrée particulièrement efficace puisque c'est

AgentM qui a remporté la compétition de l'ANAC 2014.

3.3.3 Intervalles

Dans les domaines complexes, faire une concession ne signifie pas faire une meilleure proposition du point de vue de l'adversaire : en effet, il est possible que les agents aient une utilité concordante sur une partie du domaine de négociation. Dans ce cas, une variante des stratégies basées sur les tactiques fixes consiste à choisir une proposition dans un intervalle qui ait une utilité minimale pour l'agent, tout en ayant une utilité importante pour l'adversaire. À chaque proposition qu'il doit faire, l'agent choisit parmi les propositions ayant une utilité comprise dans cet intervalle celle qui maximise l'utilité adverse. Il construit donc un intervalle $[u_{min}, u_{max}]$. La borne maximale de cet intervalle est en général l'utilité maximale du profil d'utilité de l'agent, et ne change pas au fur et à mesure de la négociation. La borne inférieure, en revanche, décroît peu à peu afin de faire de plus en plus de concession.

AgentFSEGA, développé par [Şerban *et al.*, 2012] pour des négociations où l'utilité des agents est linéaire. Il utilise ce genre de stratégies : sa borne inférieure évolue différemment en fonction de la distance à la *deadline*. il fait la distinction entre trois intervalles de temps. Sur le premier, il concède peu, et n'accepte aucune proposition. Sur le second intervalle, la borne inférieure de son intervalle descend un peu plus rapidement, de manière linéaire. Enfin la dernière partie de la négociation descend beaucoup plus rapidement, de manière exponentielle.

ValueModelAgent a été développé par [Frieder et Miller, 2013] pour des cas simples où l'utilité de l'adversaire est linéaire. Il repose lui aussi sur ce genre de techniques. Sa stratégie peut être vue comme inverse à la stratégie imitative décrite dans la Section 3.3.1. L'agent commence avec un intervalle très haut, et laisse l'adversaire faire des concessions sans changer son propre intervalle. Ce n'est que lorsque l'adversaire arrête de faire des concessions qu'il diminue sa borne inférieure, d'abord très peu lorsque la négociation en est encore à son début, puis plus rapidement à partir de 80% de la négociation, en enfin plus rapidement encore entre 90% et la fin.

3.3.4 Algorithmes génétiques

Les algorithmes génétiques prennent pour exemple le mécanisme de l'évolution des espèces : initialement, un certain nombre d'individus sont générés, puis sont croisés et mutés (modifiés). On évalue ensuite chaque individu, et les plus forts sont conservés (parmi la population initiale, les individus mutés et croisés). Ils serviront de population de base à l'itération suivante. Ces approches sont beaucoup utilisées pour les problèmes d'optimisation lorsque trouver l'optimal est complexe, mais qu'évaluer une instance est simple. Les problématiques que posent ces approches sont de représenter les instances d'un problème comme un chromosome (une section d'informations), de trouver une manière de croiser et muter un individu (voir [Eiben et Smith, 2015] pour une présentation plus complète de ces algorithmes).

En négociation automatique, les individus sont généralement des propositions générées par l'agent dans un contexte multi-attributs. Il s'agit alors de sélectionner celle qui sera la meilleure selon un critère déterminé (une fonction d'évaluation ou de *fitness*).

Parmi les agents qui se sont emparés de ces algorithmes et les ont utilisés afin de concevoir des stratégies de génération de propositions, BLGAN conçu par [Sim *et al.*, 2009] propose pour fonction d'évaluation une fonction dépendant à la fois de l'utilité de la proposition pour l'agent et de sa similitude

avec la dernière proposition faite par l'adversaire. Son agent est conçu pour un contexte mono-attribut, et la population initiale est mise à jour à chaque proposition faite par l'adversaire pour prendre en compte uniquement les propositions vues comme acceptables par l'adversaire.

GANGSTER [de Jonge et Sierra, 2016] utilise une technique similaire dans un contexte multi-attributs où les fonctions d'utilité ne sont pas linéaires. Un élément intéressant de cet algorithme génétique est sa manière de muter les vecteurs. Avant de croiser les individus, on y ajoute un vecteur aléatoire, qui sera croisé avec les dix individus sélectionnés. L'agent croise les vecteurs deux à deux et obtient ainsi 110 nouveaux vecteurs qui, ajoutés aux dix individus sélectionnés, constituent la population initiale de l'itération suivante. La fonction d'évaluation est l'utilité de l'agent. L'algorithme génétique ressort 120 propositions, parmi lesquelles l'agent choisira en fonction de la diversité par rapport à ses précédentes offres et à la distance par rapport à la dernière proposition de l'adversaire (afin d'améliorer ses chances d'acceptation). La Figure 3.8 présente chaque étape de l'algorithme, dans le cas où il y a 5 attributs.

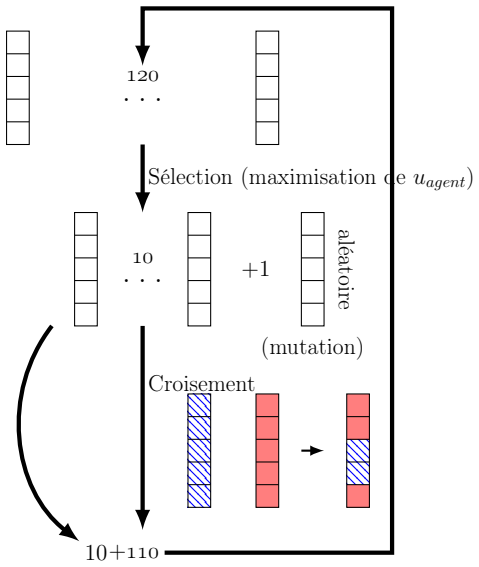


FIGURE 3.8 – Algorithme génétique de GANGSTER [de Jonge et Sierra, 2016]

Les algorithmes génétiques ont récemment fait l'objet d'une attention toute particulière, et GANGSTER n'est pas le seul à les utiliser comme base pour sa stratégie d'offres. C'est aussi le cas de k-GAgent [Kakimoto et Fujita, 2016]. k-GAgent utilise une approche peu commune : il tente de déterminer la frontière de Pareto en utilisant un algorithme génétique particulier : *Strength Pareto Evolutionary Algorithm 2* [Zitzler *et al.*, 2001]. Cet algorithme est dédié à la détermination de l'optimal de Pareto. L'algorithme demande un modèle de l'utilité de l'adversaire. Les chromosomes sont ici encore des propositions. Un ensemble de propositions *non dominées* (selon le profil d'utilité de l'agent et la modélisation de l'utilité de l'adversaire) est conservé. La fonction de sélection prend en compte cet ensemble ainsi que les individus générés en excluant ceux qui sont dominés. La taille de la population n'est donc pas fixe et varie au cours des itérations.

3.3.5 En utilisant les propositions de l'adversaire

Certains agents se sont concentrés avec succès sur les propositions faites par l'adversaire. Dans ce cadre, l'agent prend en compte la dernière ou toutes les propositions faites par l'adversaire et cherche comment les modifier pour qu'elles aient une utilité acceptable de son point de vue. Ces approches sont intéressantes parce qu'elles sont centrées sur l'adversaire, et se concentrent sur ce qui est acceptable tant pour lui que pour l'agent.

BRAM développé par [Fishel *et al.*, 2013] et CUHKAgent, conçu par [Hao et Leung, 2014] utilisent une technique de ce genre : l'agent modélise la fonction d'utilité de son adversaire. Il génère un ensemble de propositions en accord avec sa propre stratégie d'acceptation. Il recherche dans cet ensemble les propositions qui ont une utilité importante pour l'adversaire (celles qu'il est susceptible d'accepter). Il choisit enfin celle qui maximise son utilité. Si jamais il ne trouve aucune fonction qui satisfasse le critère d'acceptabilité de l'adversaire, il choisit une proposition aléatoire dans son ensemble, qui satisfait son propre critère d'acceptabilité.

GROUP2, développé par [Szöllősi-Nagy *et al.*, 2016] utilise une stratégie proche quoique différente. Il reprend la proposition de l'adversaire et cherche à l'améliorer. Il commence par optimiser son propre bien-être, puis lorsqu'il s'approche de la deadline, le bien être social des deux agents. Il utilise une variante gloutonne du Coordinate Descent où l'attribut amenant la plus grande amélioration du critère retenu (utilité ou bien-être) est choisi. On sélectionne la meilleure valeur pour ce critère. Ces étapes sont ensuite réitérées avec les autres attributs.

3.3.6 Discussion

Les stratégies de proposition ont beaucoup évolué au fur et à mesure que l'on considérait des domaines de de négociation de plus en plus proches de la réalité. Il est intéressant de constater que des stratégies inspirées des stratégies fixes, conçues sur une base purement heuristique, tiennent toujours le haut du classement.

Certaines stratégies issues d'autres domaines de l'Intelligence Artificielle, tels que les algorithmes génétiques, ont fait leur apparition parmi les finalistes. Il est surprenant que d'autres techniques, pourtant utilisées par ailleurs en IA n'aient pas été testées dans ce contexte. Les algorithmes à base d'arbre de jeux sont complexes à mettre en œuvre dans un contextes où les attributs sont continus, mais certaines solutions ont été proposées avec succès par le passé. De même, il paraît surprenant que les méthodes de Monte Carlo, qui ont récemment remporté énormément de succès, ne figurent pas parmi les techniques utilisées en négociation automatique.

Cela est peut-être dû au fait que l'ANAC a servi de base pour les benchmarks des agents de négociation. Or, la compétition fixe le temps de négociation à 3 minutes, et fournit des ressources de calcul limitées. C'est dans ce cadre que nous proposons, au Chapitre 5 de voir quelles sont les capacités d'un agent de négociation basé sur une recherche arborescente de Monte-Carlo, tant en termes de performance de négociation, qu'en termes d'adaptativité.

3.4 Stratégie d'acceptation

La stratégie d'acceptation d'un agent est la fonction qui lui indique s'il doit ou non accepter la proposition faite par un adversaire. Accepter une proposition peut être exprimé comme un pari : l'agent accepte de recevoir l'utilité qui lui est proposée en pariant que les propositions ultérieures ne lui amèneront pas une utilité supérieure. Pour un agent qui ne connaît pas la stratégie d'offres de l'adversaire, la stratégie d'acceptation ne peut être qu'heuristique, puisqu'il ne peut pas anticiper les futures propositions qui lui seront faites. La majorité des agents utilisent des stratégies de ce type, simples. Mais là encore, plusieurs tentatives ont été faites pour concevoir des stratégies de plus en plus complexes. Cependant, les stratégies utilisées restent moins nombreuses et variées que les stratégies d'offres qui ont vu l'essor de nouveaux types d'algorithmes.

3.4.1 Stratégies simples

La première stratégie simple consiste à fixer un seuil au-dessus duquel les agents acceptent une proposition à quelque moment que ce soit. Cette stratégie a été utilisée par un certain nombre d'agents, parmi lesquels [Fishel *et al.*, 2013, Dirkzwager *et al.*, 2013]. Bien que cette stratégie se soit montrée gagnante dans certains cas, elle n'est pas exempte de défauts. En particulier, elle demande de déterminer le seuil au-delà duquel l'agent est prêt à accepter une proposition. Cela est notamment possible lorsque l'agent a une connaissance du domaine. C'est par exemple ce que révèlent [Zafari et Nassiri-Mofakham, 2016] : leur stratégie consiste en une variante de cette stratégie (le seuil évoluant en fonction de la proximité de la borne). Le seuil a été choisi en analysant les domaines de négociation des précédentes compétitions. Bien que cette stratégie se soit avérée payante pour nombre d'agents, elle est complexe à mettre en œuvre dans un contexte réaliste où le domaine de négociation est totalement inconnu.

Une autre stratégie consiste à vérifier si la dernière proposition reçue est meilleure que celle qui a été produite par la stratégie d'offres. Si tel est le cas, l'agent accepte la proposition adverse. Le principe de cette stratégie est qu'au lieu de faire une proposition qui sera pire que celle qu'il vient de recevoir, l'agent préférera accepter celle qu'il a reçue. C'est la stratégie d'acceptation proposée par [Faratin *et al.*, 1998]. Cette stratégie a elle aussi été utilisée par [Erez et Zuckerman, 2016]. Une variante a été proposée par [Niimi et Ito, 2016] : l'agent accepte la proposition adverse si celle-ci est meilleure qu'une de celles qui ont été proposées par l'agent lui-même. Le principe reste le même, mais il est appliqué de manière plus étendue. Cette stratégie d'acceptation n'est cohérente que si l'agent se conforme aussi à ce principe dans sa stratégie d'offres et qu'il procède à des concessions à chaque tour (typiquement, avec une stratégie basée sur le temps telle que décrite dans la Section 3.3.1).

Il est possible d'éviter le calcul d'une proposition si, au lieu de comparer la dernière proposition reçue avec celle qui est produite par la stratégie d'offres, l'agent compare la proposition de l'adversaire avec la dernière qu'il a lui-même produite. Cette stratégie est notamment utilisée par AgentSmith ([van Galen Last, 2012]).

Ces stratégies ont été largement utilisées par les agents lors de l'ANAC, mais certains auteurs ont conçu des stratégies plus complexes, qui sont présentées dans les sections suivantes.

3.4.2 Combinaison de stratégies simples

La première combinaison consiste à changer de condition d'acceptation en fonction du stade de la négociation. [An et Lesser, 2012, Zafari et Nassiri-Mofakham, 2016, Şerban *et al.*, 2012] distinguent plusieurs stades de négociation, et acceptent plus facilement lorsque la négociation arrive à la fin, ou bien ils modifient leur seuil d'acceptation. Certains d'entre eux, [Zafari et Nassiri-Mofakham, 2016] par exemple, acceptent même automatiquement la dernière proposition qui leur a été faite, afin d'éviter de recevoir l'utilité de réserve.

[Baarslag *et al.*, 2014] proposent de combiner trois types de stratégies afin d'en obtenir une plus adaptée à l'adversaire. Les auteurs reprennent les stratégies simples énoncées ci-dessus. Ils prennent également en compte la stratégie de seuil et une dernière stratégie qui accepte la proposition d'un adversaire une fois que la *deadline* est assez proche. La stratégie combinée est alors : soit la proposition de l'adversaire est meilleure que la prochaine proposition de l'agent, soit la *deadline* est proche et la proposition de l'adversaire a une utilité au-dessus d'un seuil. Les auteurs montrent que leur stratégie est meilleure que les stratégies simples présentées dans la Section 3.4.1. Cette stratégie a été reprise notamment par [Baarslag *et al.*, 2013, Koeman *et al.*, 2015].

3.4.3 Stratégie optimale

[Baarslag *et al.*, 2013] proposent une stratégie d'acceptation optimale. Ils font pour cela une assertion sur leur adversaire : celui-ci est supposé suivre une fonction connue (mais bruitée). Les auteurs prennent en compte 3 modèles de stratégie, de la plus simple à la plus complexe :

RandomWalker La fonction de stratégie de cette agent est totalement aléatoire (bruit blanc).

Stratégie fixe, connue Il s'agit d'une fonction fixe dépendant du temps, telle que définie dans la Section 3.3.1, mais avec une composante aléatoire.

Stratégie inconnue Pour connaître sa stratégie, Baarslag propose de modéliser l'adversaire, par exemple en utilisant la régression de processus gaussien, comme proposé par [Williams *et al.*, 2011].

Un fois cette information obtenue, les auteurs proposent d'accepter de manière optimale : en prenant en compte le nombre de propositions qu'il reste à recevoir, ils calculent les probabilités d'obtenir une meilleure offre, ce qui leur permet d'optimiser l'utilité moyenne obtenue.

3.4.4 Courbe de concession

Une stratégie intéressante a été introduite par [Hao et Leung, 2014]. Leur agent prend en compte plusieurs caractéristiques de l'adversaire et détermine un paramètre de temps en fonction de ce profil, du scénario de négociation et de la *deadline*. L'agent découpe ensuite la session en deux parties, et crée une courbe de concession pour chacune. Il accepte une proposition si celle-ci se trouve au-dessus de la courbe créée. La première courbe concède très peu, et les conditions d'acceptation y sont donc particulièrement difficiles. La seconde a une pente plus douce, ce qui permet de faire davantage de concessions. Le paramètre est mis à jour au fur et à mesure de la négociation, permettant de l'affiner en fonction des informations obtenues sur l'adversaire.

Une approche assez similaire a été développée par [Kawaguchi *et al.*, 2012, Kawaguchi *et al.*, 2013]. Cet agent établit une probabilité d'acceptation, qui dépend de la proposition de l'adversaire, ainsi que

de son modèle, ou plus précisément de l'utilité maximale attendue de sa part. Plus l'offre reçue est près de cette offre, et de l'objectif que s'est fixé l'agent en conséquence, plus la probabilité est importante. Elle augmente aussi lorsqu'on s'approche de la fin de la négociation.

3.4.5 Discussion

Les approches proposées sont beaucoup moins nombreuses en ce qui concerne les stratégies d'acceptation qu'en ce qui concerne la modélisation d'adversaires, ou même les stratégies de proposition. Cela est peut-être dû au fait que les stratégies simples, que nous avons présentées en Section 3.4.1 semblent être efficaces. Les gagnants de l'ANAC s'en satisfont souvent, et avec succès.

Les stratégies combinées ont elles aussi eu du succès, puisque FawkesAgent ([Koeman *et al.*, 2015]) qui a gagné l'ANAC 2013 l'utilise. Les autres stratégies ont été peu ou pas utilisées en raison de la complexité de leur mise en place par rapport aux avantages qu'elles procurent.

3.5 Conclusion

La création de l'ANAC a eu une influence déterminante sur le développement de la négociation automatique. Elle a eu pour effet de créer un dynamisme qui a été à l'origine des nombreux travaux que nous avons rassemblés dans cette section. Cette profusion a cependant un revers : l'ANAC et la librairie sur laquelle elle s'appuie, GENIUS ([Lin *et al.*, 2014]) ont servi comme outil de comparaison privilégié sinon exclusif. Or, il est utile de rappeler que l'ANAC est une compétition avec certaines règles, et que ces règles ne sont pas applicables à tous les contextes. Il en va ainsi des protocoles par exemple. L'ANAC a proposé le *bargaining* comme principal protocole, avant de s'intéresser à la négociation tripartite et aux smart grids. Mais d'autres protocoles de négociation existent, comme nous l'avons décrit dans le Chapitre 2.

Et l'ANAC fait aussi d'autres suppositions qui sont justifiées dans son contexte, mais qui ne sont pas nécessairement applicables dans l'ensemble des situations réelles. Il en va ainsi pour la présence d'une *deadline* : GENIUS fait la supposition qu'il existe une limite à la négociation, qui s'exprime en nombre de tours ou en temps. Bien que cette contrainte semble justifiée dans certains cas, il est possible qu'elle se montre caduque dans certains contextes. On pourrait alors argumenter que chaque agent est doté de sa propre *deadline*, et que celle-ci peut être apprise en utilisant les approches décrites dans la Section 3.2.5. Mais cela suppose que l'adversaire a au moins une *deadline* privée, décidée lorsqu'il commence la négociation. Or, ce n'est pas forcément le cas. Il est possible qu'un agent commence la négociation sans avoir aucune idée du temps qu'il compte passer à négocier. Il est dans ce cas impossible de déduire la *deadline* de l'adversaire et de l'utiliser pour concevoir une stratégie.

GENIUS a été prévu pour la mise en place des attributs continus, mais cette fonctionnalité n'a pas encore été implémentée. Les attributs continus sont donc représentés sous la forme d'entier. Or, les entiers sont dénombrables et un intervalle entier est fini, ce qui n'est pas le cas d'un intervalle réel. Certains composants reposent sur cette fonctionnalité : il est par exemple possible de calculer une fréquence pour chaque valeur d'un attribut entier, ce qui est impossible pour un attribut continu.

Ces problématiques forment un objectif intéressant pour la négociation automatique, et qui pourraient peut-être l'aider à s'implanter davantage dans les applications industrielles, où les suppositions qui sont faites sur la négociation sont encore trop restrictives pour être applicables dans certains

contextes. Construire un tel agent est donc un enjeu essentiel de la négociation automatique, et un défi intéressant pour les chercheurs dans ce domaine.

Chapitre 4

Agents curieux

“
We might have known from the first that human curiosity is undying, and that the results we announced would be enough to spear others ahead on the same age-long pursuit of the unknown.
Howard Phillips LOVECRAFT, *At the Mountains of Madness*, 1931

”

Sommaire

4.1 Introduction 94
4.2 Sensibilité à la curiosité 94
4.2.1 Problématique 95
4.2.2 Formalisation 95
4.2.3 Taxonomie des agents 98
4.2.4 Discussion 101
4.3 Protocole résistant aux agents curieux 101
4.3.1 *Bargaining* et agents curieux 101
4.3.2 Cas mono-attribut : négociation sur le prix 102
4.3.3 Cas multi-attributs 107
4.3.4 Discussion 111
4.4 Implémentation 111
4.4.1 Implémentation centralisée 111
4.4.2 Implémentation décentralisée 112
4.4.3 Implémentation 1:n 115
4.4.4 Analyse expérimentale 116
4.4.5 Discussion 121
4.5 Conclusion 121

4.1 Introduction

Imaginons la situation suivante : vous négociez avec un marchand en échangeant offres et contre-offres – en suivant le protocole de *bargaining*. Vous souhaitez acheter son bien et lui souhaite le vendre, chacun en optimisant sa propre utilité. Supposons maintenant que le marchand ait connaissance du prix maximum auquel vous accepteriez d'acheter son bien. Vous seriez alors en droit de vous demander comment le marchand a acquis cette information sur vous. La réponse à cette question tient dans deux éléments : le premier, c'est l'*opponent modeling*, que nous avons présenté à la Section 3.2. L'*opponent modeling* permet notamment d'inférer l'utilité et le prix de réserve d'un agent. Comme nous l'avons souligné, ces techniques reposent sur l'apprentissage automatique, et nécessitent que l'agent soit capable d'accumuler assez de données pour que l'apprentissage soit efficace. Le second élément concerne le protocole utilisé dans la négociation. En effet, le *bargaining* classique est un protocole sans intermédiaire, qui repose sur un échange d'informations entre les deux agents. En outre, il est possible à un agent de se désengager de la négociation sans pénalité, ce qui lui permet d'accumuler des informations sans avoir à payer.

Nous qualifions de « curieux » un agent exploitant les particularités du *bargaining* pour acquérir des informations. Les agents curieux sont conscients de la valeur de l'information, et cherchent à en acquérir en négociant avec les autres agents. Cette particularité a des conséquences sur l'incitation qu'ils ont à trouver un accord avec leur adversaire : à défaut d'aboutir, la négociation peut leur apporter des informations – les propositions faites par l'adversaire – qui pourront être utilisées pour le modéliser. Si ce type de comportement est nuisible pour le système, il l'est encore davantage pour les agents qui, conscients de l'importance qu'ont leurs informations, souhaitent les garder secrètes.

Ce chapitre a deux objectifs : nous proposons d'abord de caractériser formellement les agents curieux et discrets, en introduisant un modèle d'utilité qui ne repose pas uniquement sur le prix obtenu à la fin de la négociation mais aussi sur les informations révélées et obtenues au cours de celle-ci. Nous proposons ensuite la conception d'un protocole de *bargaining* basé sur la présence d'un tiers de confiance. Notre protocole, bien que basé sur la présence un médiateur, diffère de celui de [Myerson et Satterthwaite, 1983] et de celui de [Osborne et Rubinstein, 1990] où la négociation est gérée au plus près par le médiateur, les agents étant réduits au mieux à accepter ou refuser sa proposition. Dans notre protocole, les agents font eux-mêmes des propositions, et le médiateur ne tient qu'un rôle de contrôle afin d'éviter la révélation inutile d'informations. Il n'en reste pas moins que les agents révèlent leurs informations privées au médiateur et que le protocole rencontre donc une problématique commune, à savoir s'assurer que les informations confiées au médiateur sont bien les bonnes (que le protocole est incitatif, comme décrit dans la Définition 2.13).

4.2 Sensibilité à la curiosité

Une des problématiques les plus centrales en négociation automatique est l'absence d'information sur l'adversaire. En effet, comme le font remarquer [Baarslag *et al.*, 2015b] « Bien que le fait de partager des informations puisse se solder par des gains mutuels, les négociateurs sont peu enclins à partager des informations dans un contexte ayant un aspect compétitif afin d'éviter leur exploitation par l'autre partie » [1]. Dans ce chapitre, nous nous intéressons aux abus que ces possibilités peuvent amener. Il est en effet possible d'acquérir de l'information sur l'adversaire en utilisant des techniques d'apprentissage

1. « *Despite the fact that sharing private information can result in mutual gains, negotiators are unwilling to share information in situations with a competitive aspect to avoid exploitation by the other party* »

automatique. Ces méthodes, que nous avons décrites en Section 3.2, reposent sur l'exploitation des données acquises lors des échanges qu'implique la négociation. Elles permettent à l'agent qui possède le meilleur modèle et la plus grande puissance de calcul de mieux exploiter les informations acquises sur son adversaire et de tirer profit de la négociation, potentiellement à ses dépens. Dans cette section, nous caractérisons trois classes d'agents en fonction de leur rapport à l'information acquise et révélée : les agents curieux, incurieux, et discrets.

4.2.1 Problématique

La curiosité est une qualité importante pour être un bon négociateur. Cependant, dans le contexte de la négociation automatique, se concentrer sur la collecte d'informations pour modéliser son adversaire devient un avantage démesuré. Parce qu'il est si manipulable, le protocole de *bargaining* pousse parfois les agents honnêtes à entrer dans de longues négociations ou à n'en initier aucune afin d'y échapper.

A notre connaissance, les économistes n'ont pas proposé de modélisation de la curiosité qui puisse être exploitée dans les négociations automatiques. De plus, aucune recherche n'a été menée pour rendre le *bargaining* robuste aux comportements curieux.

On définit un agent curieux comme un agent qui cherche à négocier non seulement pour obtenir des biens, mais aussi pour obtenir des informations. Ces informations étant par la suite utilisées pour nourrir un modèle, la quantité d'informations obtenue est critique pour modéliser l'adversaire aussi précisément que possible. Or, comme le souligne [Baarslag *et al.*, 2015b], certains agents souhaitent conserver certaines informations privées. Ils ne souhaitent pas, en particulier, que les négociations menées avec l'autre partie lui servent de données pour une modélisation, lui permettant ainsi d'inférer des informations qu'ils veulent garder privées.

4.2.2 Formalisation

En outre si le protocole de négociation le permet, les agents curieux peuvent amener des comportements néfastes dans la négociation :

- un allongement artificiel du nombre d'échanges de propositions, l'agent curieux faisant son possible pour ralentir la négociation et ainsi acquérir plus d'informations,
- un grand nombre de négociations n'aboutissant pas, puisque l'agent curieux peut poursuivre une négociation qui ne l'intéresse pas uniquement pour obtenir des informations sur l'autre partie et enfin,
- un déséquilibre entre les acteurs, en particulier lorsque ceux-ci n'ont pas les mêmes moyens, et c'est précisément dans ce cas que nous nous trouvons : d'un côté, les entreprises qui font financer leur BFR sur notre plate-forme sont des PME en manque de liquidité, n'ayant pas les moyens d'investir dans des techniques de *machine learning* complexes et coûteuses. De l'autre, les investisseurs sont en surplus de liquidité et peuvent voir ces techniques comme un moyen d'obtenir de meilleurs prix.

La négociation a été souvent associée au concept de jeu (au sens de théorie des jeux). Nous avons vu dans la Section 2.3 une définition du *bargaining* comme un mécanisme sur un jeu bayésien stratégique. Ce modèle est utilisé par [Nash, 1950, Myerson, 1979] parce que les agents ne négocient pas à proprement parler : ils confient leurs préférences à un médiateur qui choisit pour eux une solution. Comme nous l'avons exposé, notre protocole laisse le soin aux agents de négocier par eux-mêmes de manière itérée. Les agents curieux s'appuient précisément sur le caractère

itéré de la négociation pour acquérir des informations sur leur adversaire. Or, les jeux normaux ne sont pas conçus pour modéliser de telles situations. La modélisation qui leur correspond le mieux n'est donc pas une modélisation sous forme de jeu normal mais sous forme de jeu extensif (voir [Osborne et Rubinstein, 1994, Shoham et Leyton-Brown, 2008b]). La relation entre *bargaining* classique et théorie des jeux a notamment été établie par [Kuhn, 1962, Brams, 2003].

Nous allons donc donner une définition du *bargaining* comme jeu extensif. À cette fin, nous commençons par définir les jeux extensifs. Un jeu extensif ou séquentiel est un jeu où les joueurs jouent les uns après les autres, *i.e.* à chaque moment, c'est à un et un seul joueur d'accomplir une action. Ces jeux sont caractérisés par un ensemble d'historiques possibles, qui sont les séquences d'actions qui peuvent être réalisées par l'ensemble des joueurs selon les règles du jeu. En fait, il est possible de caractériser un jeu par ces historiques, ce qui évite de définir des règles qui dépendront du jeu lui-même : cette définition à partir des historiques possibles est très générale et permet de définir n'importe quel jeu extensif. Notons que selon leurs règles, certains jeux peuvent être infinis (la séquence d'actions réalisées par les joueurs ne se termine pas). En 1994, Osborne et Rubinstein ont formalisé ces éléments et ont abouti à la définition de jeu extensif (à information parfaite, *i.e.* l'historique des actions est connu).

Définition 4.1 (Jeu sous forme extensive)
Un jeu extensif peut être caractérisé par quatre éléments ([Osborne et Rubinstein, 1994]) :

1. *l'ensemble des joueurs, représentés par des naturels de 1 à N (nombre total de joueurs),*
2. *l'ensemble des séquences possibles d'actions (aussi appelées historiques) H. Cet ensemble a trois propriétés*
   (a) *historique initial : $\varnothing \in H$,*
   (b) *antériorité : si un historique possible (i.e. il est possible selon les règles du jeu de jouer chaque action qui le compose dans cet ordre), alors toute sous-suite constituée des L premiers éléments de cet historique constitue aussi un historique possible*

$$\forall K \in \mathbb{N}, \forall L < K, \left(a^i\right)_{i \leq K} \in H \implies \left(a^i\right)_{i \leq L} \in H \tag{4.1}$$

   (c) *extension à l'infini : s'il existe une suite croissante d'historiques au sens où chaque historique de cette suite est constitué du précédent suivi d'une nouvelle action, et que cette suite n'est pas bornée, alors, la suite infinie qui correspond au passage à la limite de cette suite d'historique, et qui est un historique infini, est aussi possible*

$$\left[\forall L \in \mathbb{N}, \left(a^i\right)_{i \leq L} \in H\right] \implies \left(a^i\right)_{i \in \mathbb{N}} \in H \tag{4.2}$$

   Chaque élément de H est un historique, et chaque élément de chaque historique est un coup joué par un joueur. Un historique est dit terminal ssi il est infini, ou s'il y a un naturel K tel que : $\left(a^i\right)_{i \leq K}$ est dans H et pour tout a action dans le jeu, $(a_1, \ldots, a_K, a)$ n'est pas dans H. L'ensemble des historiques terminaux est appelé Z,
3. *une fonction joueur qui à chaque historique non terminal associe le joueur qui doit jouer,*
4. *pour chaque joueur un ordre sur les historiques terminaux ($\succ_i$). Pour un joueur i et deux situations finales d'un jeu i.e. deux historiques terminaux $(z, z') \in Z^2$, $z \succ_i z'$ signifie que le joueur i préfère la situation z à la situation z'.*

Définition 4.2 (*Bargaining*)
En s'appuyant sur cette définition, il est possible de définir une négociation $\mathcal{B}$ comme un quadruplet $\mathcal{B} = \left(N, H, A, (u_i)_{i \in [\![0,N]\!]}\right)$:

1. *il y a deux joueurs : un acheteur et un vendeur. On a donc* $N = 2$. *Sans perte de généralité, on considère que l'acheteur est le joueur* 1 *et que le vendeur est le joueur* 2,
2. *l'historique est composé des messages que les agents s'envoient l'un à l'autre : les propositions faites par les agents, ainsi que les autres messages possibles (*`accept`, `reject`*). Les historiques terminaux sont les historiques infinis et les historiques se terminant par* `accept` *ou par* `reject`*. Chaque message est composé d'un quadruplet* (n_1, n_2, α, c) *où* n_1, n_2 *sont respectivement l'expéditeur et le destinataire du message.* α *est l'acte de langage (speech act) du message :* $\alpha \in \{\texttt{accept}, \texttt{reject}, \texttt{propose}\}$. *Enfin,* c *est le contenu du message. il s'agit d'une liste de couples* (k, v) *où* k *est la clé d'un attribut du domaine de négociation et* v *la valeur correspondante. On propose, de plus, de séparer tout historique en deux historiques correspondant aux actions de chaque joueur. Ainsi :* $h_i = \{(n_1, n_2, \alpha, c) \in h | n_1 = i\}$,
3. *la fonction joueur fonctionne selon la parité. On suppose que c'est l'acheteur (le joueur 1) qui commence. Ainsi,* $\forall h \in H, joueur(h) = |h| \mod 2 + 1$. $|h|$ *est la taille de l'historique* h. *Si aucun message n'a encore été échangé, c'est au joueur 1 (l'acheteur) de jouer. Si 9 messages ont été échangés, c'est au joueur* $9 \mod 2 + 1 = 2$ *de jouer,*
4. *L'ordre sur les situations terminales de chaque joueur est induit par une fonction d'utilité. Ces fonctions d'utilité, notées* u_i *sont des fonctions qui associent l'utilité la « satisfaction » de l'agent à un historique terminal :* $\forall i \in \{1, 2\}, u_i \in \mathbb{R}^Z$. *On considère en outre et sans perte de généralité que l'utilité d'un agent est nulle si l'agent ne prend pas part à une négociation.*

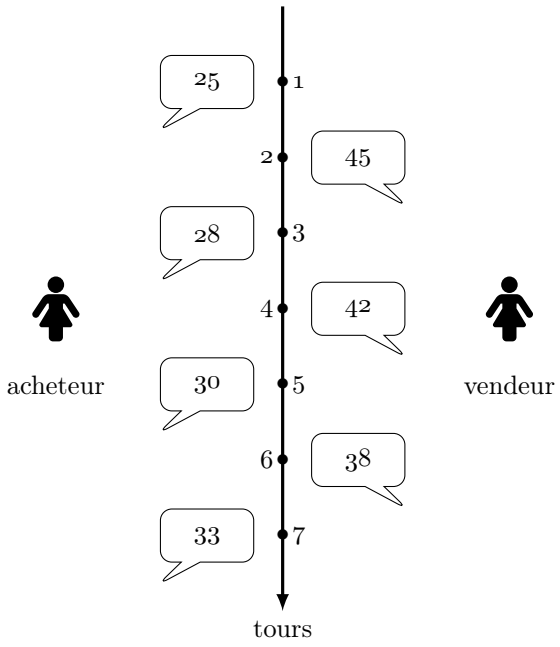


FIGURE 4.1 – Exemple de négociation

Afin de clarifier cette définition, nous donnons un exemple de négociation sur la Figure 4.1. Dans cette figure, un acheteur (1) et un vendeur (2) échangent des propositions de prix pour un objet. L'historique n'est pas terminal puisque aucun des agents n'a envoyé un `reject` ou un `accept`. L'historique est : $([1, 2, \texttt{propose}, 25],\ [2, 1, \texttt{propose}, 45],\ [1, 2, \texttt{propose}, 28],\ [2, 1, \texttt{propose}, 42],\ [1, 2, \texttt{propose}, 30],\ [2, 1, \texttt{propose}, 38],\ [1, 2, \texttt{propose}, 33])$. Comme l'historique n'est pas final, il est possible de définir la

fonction *joueur* : $joueur(h) = |h| + 1 \mod 2 = 7 + 1 \mod 2 = 2$. C'est donc au joueur 2, *i.e.* au vendeur de jouer.

On considère généralement que la fonction d'utilité des agents ne dépend que du dernier message envoyé, et non de la séquence de messages dans son ensemble. Pourtant, comme nous l'avons exposé précédemment, afin d'obtenir autant d'informations que possible sur la négociation, l'utilité des agents peut tout à fait dépendre pas seulement de cette dernière action, mais aussi de l'ensemble de l'historique. Dans la suite de cette section, nous introduisons des fonctions d'utilité qui dépendent également de la séquence de propositions que les agents ont reçues.

4.2.3 Taxonomie des agents

Dans cette section, nous caractérisons quatre types d'agents, selon qu'ils souhaitent ou non protéger leurs propres informations privées et selon s'ils sont désireux ou non d'obtenir des informations sur l'adversaire. Nous décrivons comment leur utilité varie avec les attributs du dernier message reçu (typiquement le prix), et l'information divulguée et collectée (l'historique de la négociation). Ces quatre types d'agents sont les agents incurieux, strictement curieux, strictement discrets et enfin curieux et discrets. Nous introduisons deux ensembles de comparateurs indiquant si leur utilité est affectée par une telle variation.

- $\succ_a^\pi$ **et** $\sim_a^\pi$ sont respectivement des relations d'ordre strict et d'équivalence sur les attributs. Ils permettent de représenter les préférences d'un acheteur ou un vendeur sur les attributs
- $\succ_a^h$ **et** $\sim_a^h$ sont des relations d'ordre strict et d'équivalence sur une séquence de propositions (V). Pour deux séquences de propositions h et h', $h \succ_a^h h'$ (resp. $h \sim_a^h h'$) signifie que selon l'agent i, h amène plus (resp. autant) d'information que h'

Afin d'étendre le modèle classique de rationalité de façon cohérente, nous supposons que, toutes choses égales par ailleurs (*ceteris paribus*), un agent préfère une négociation s'achevant sur une meilleure proposition pour lui. En d'autres termes, si deux négociations révèlent la même quantité d'information, un agent a préfère celle qui optimise la proposition. L'équation suivante est donc vérifiée par n'importe quel agent rationnel :

$$\forall (b, b') \in \mathcal{B}^2, \begin{cases} \pi_b \succ_a^\pi \pi_{b'} \\ h_b^2 \sim_a^h h_{b'}^2 \\ h_b^1 \sim_a^h h_{b'}^1 \end{cases} \implies u_a(h_b) > u_a(h_{b'}) \tag{4.3}$$

et

$$\forall (b, b') \in \mathcal{B}^2, \begin{cases} \pi_b \sim_a^\pi \pi_{b'} \\ h_b^2 \sim_a^h h_{b'}^2 \\ h_b^1 \sim_a^h h_{b'}^1 \end{cases} \implies u_a(h_b) = u_a(h_{b'}) \tag{4.4}$$

où h_b est l'historique de la négociation b, et h_b^2 (resp. h_b^1) est l'historique de la négociation b réduit aux actions du vendeur (resp. de l'acheteur).

Les agents sensibles à la curiosité considèrent l'information dévoilée et/ou collectée. Nous les caractérisons donc sur deux axes. D'une part, les agents *curieux* (par opposition aux agents *incurieux*) ont une utilité à collecter des information sur leur adversaires à des fins d'*opponent modelling*. D'autre part, les agents *discrets* (par opposition aux agents non-discrets) ont une aversion à la divulgation

d'information car ils ont conscience de l'existence des agents *curieux*. Ces axes étant orthogonaux, un agent sensible à la curiosité est une combinaison de ces deux types.

Définition 4.3 (Agent incurieux)
Un agent incurieux est un agent du modèle classique du marchandage. Son seul intérêt est l'issue de la négociation.

L'utilité d'un agent incurieux n'est pas affectée par l'information échangée.

$$\forall (b, b') \in \mathcal{B}^2, \pi_b \succ_a^\pi \pi_{b'} \implies u_a(h_b) > u_a(h_{b'}) \tag{4.5}$$

Définition 4.4 (Agent strictement discret)
Un agent strictement discret (discret et incurieux) sait que ses actions révèlent de l'information qui pourrait être utilisée par des agents malicieux à ses dépens. Il n'est cependant intéressé que par l'achat ou la vente de biens et pas par la collecte d'information. Les concessions qu'il fait sur les propositions qu'il est prêt à accepter pour un bien décroissent avec le nombre de propositions. Ainsi, il tend à abandonner la négociation plus tôt qu'un agent classique (incurieux). De plus, si la négociation échoue, son utilité est négative, puisqu'il a révélé des informations et qu'il n'a rien obtenu en retour.

Un agent discret préfère ne pas révéler d'information, mais n'est pas intéressé par la collecte d'informations non plus. Pour l'agent i :

$$\forall (b, b') \in \mathcal{B}^2, \begin{cases} \pi_b \sim_i^\pi \pi_{b'} \\ h_{b'}^i \succ_i^h h_b^i \end{cases} \implies u_i(h_b) > u_i(h_{b'}) \tag{4.6}$$

Définition 4.5 (Agent strictement curieux)
Un agent strictement curieux (curieux et non discret) est intéressé par la collecte des informations qu'il pourra utiliser pour modéliser son adversaire. Il ne s'inquiète pas de l'information qu'il révèle. Les concessions qu'il fait sur les propositions qu'il est prêt à accepter pour un bien augmentent avec la quantité d'information collectée. Il a une utilité positive en cas de rejet. Une bonne stratégie pour lui consiste donc à faire durer la négociation autant que possible puis la rejeter. Un agent curieux cherche à collecter autant d'information que possible pendant un échange, mais ne s'inquiète pas de l'information qu'il dévoile. Dans le cas d'un agent i (on note son adversaire $-i$) :

$$\forall (b, b') \in \mathcal{B}^2, \begin{cases} \pi_b \sim_i^\pi \pi_{b'} \\ h_b^{-i} \succ_i^h h_{b'}^{-i} \end{cases} \implies u_i(h_{b'}) > u_i(h_b) \tag{4.7}$$

Définition 4.6 (Agent curieux et discret)
Un agent curieux discret fait de la modélisation d'adversaire, et cherche à se préserver d'une telle modélisation. Il implémente l'Équation (4.8)*, une version plus contrainte des Équations* (4.6) *et* (4.7) *(notées* (4.6′) *et* (4.7′)*). Il cherche à collecter des informations ceteris paribus, et à en révéler le moins d'information possible ceteris paribus. Pour un agent i :*

$$\forall (b, b') \in \mathcal{N}^2, \begin{cases} \overset{(4.6')}{\pi_b \sim_i^\pi \pi_{b'}} \\ h_b^{-i} \sim_i^h h_{b'}^{-i} \\ h_{b'}^i \succ_i^h h_b^i \end{cases} \vee \begin{cases} \overset{(4.7')}{\pi_b \sim_i^\pi \pi_{b'}} \\ h_b^i \sim_i^h h_{b'}^i \\ h_b^{-i} \succ_i^h h_{b'}^{-i} \end{cases} \implies u_i(h) > u_i(h') \tag{4.8}$$

La Figure 4.2 décrit des exemples de fonctions de prix de réserve pour ces types d'agents que nous détaillons ci-après dans le cadre d'une négociation mono-attribut (l'unique attribut est le prix). Du fait que le *bargaining* est un protocole itératif où les agents envoient des messages l'un après l'autre, l'agent discret et curieux peut avoir trois types de comportements :

1. lorsque sa sensibilité curieuse et sa sensibilité discrète s'annulent, il se comporte comme un agent indiscret, comme lors des premiers tours sur la Figure 4.2,
2. lorsque sa sensibilité discrète l'emporte sur sa sensibilité curieuse, il se comporte comme un agent discret, comme entre les tours 4 et 14 et à partir du tour 34
3. lorsque sa sensibilité curieuse est plus forte que sa sensibilité discrète, il se comporte comme un agent curieux, comme entre les tours 14 et 34.

Pour ces raisons, il est assimilé dans la suite de ce chapitre à un des trois types élémentaires d'agents, et n'est pas étudié séparément.

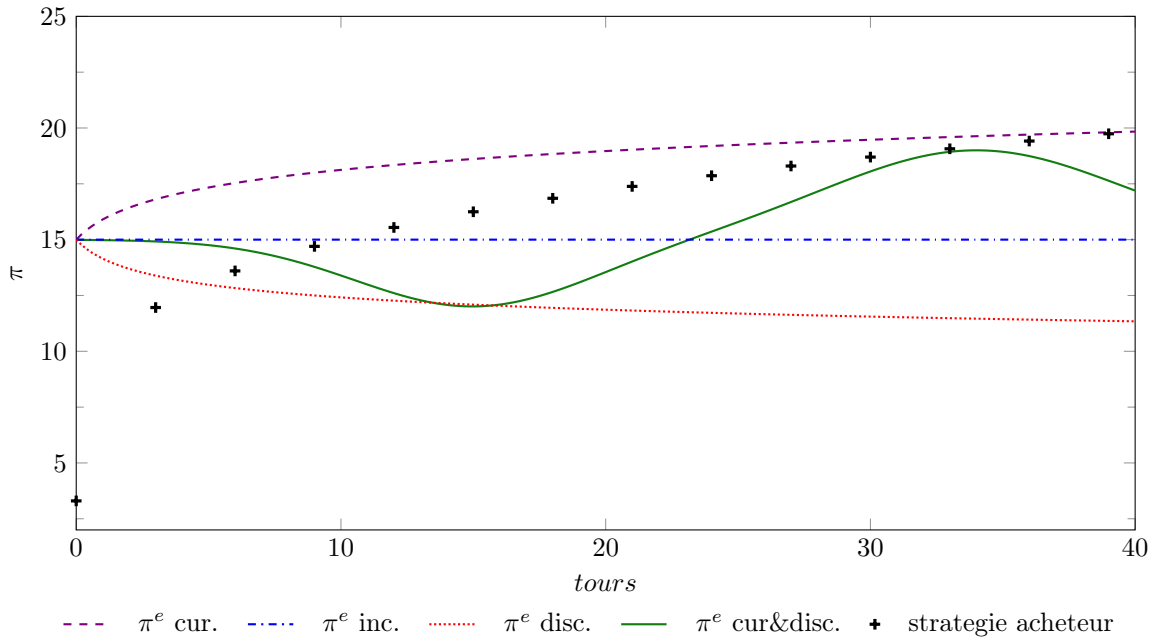


FIGURE 4.2 – Stratégie et prix de réserve des différents types d'agents

On déduit de ces définitions les deux théorèmes suivants, sur les agents strictement curieux et strictement discrets. Ces théorèmes ne sont pas applicables aux agents curieux et discrets. Par définition, un agent incurieux recevra une utilité nulle en cas de rejet.

Théorème 4.1 (Utilité d'un agent strictement discret après un `reject`)
Soit a un agent discret. Si $u_a(\varnothing) = 0$, alors $\forall h \notin Z$, $u_a(h; reject) \leq 0$.

Démonstration. $\forall h \notin Z, h \succ_a^h \varnothing$ ou $h \sim_a^h \varnothing$. De plus $h \sim_a^\pi \varnothing$. Donc, selon la Définition 4.4 et l'Équation (4.4), $u_a(h) < u_a(\varnothing) = 0$ ou $u_a(h) = u_a(\varnothing) = 0$. □

Théorème 4.2 (Utilité d'un agent strictement curieux après un `reject`)
Soit a un agent curieux. Si $u_a(\varnothing) = 0$, alors $\forall h \notin Z$, $u_a(h; reject) \geq 0$.

Démonstration. $\forall h \notin Z, h \succ_a^h \varnothing$ ou $h \sim_a^h \varnothing$. De plus $h \sim_a^\pi \varnothing$. Donc, selon la Définition 4.5 et l'Équation (4.4), $u_a(h) > u_a(\varnothing) = 0$ ou $u_a(h) = u_a(\varnothing) = 0$. □

4.2.4 Discussion

En donnant les définitions des agents discrets et curieux, on voit déjà apparaître une inégalité structurelle entre ces deux types d'agents. Là où les agents strictement curieux obtiennent une utilité positive ou nulle en quittant une négociation sans accord, les agents strictement discrets obtiennent une utilité négative. Cette inégalité peut pousser les agents curieux à négocier en sachant dès le départ qu'ils rejetteront tout accord final, la négociation ayant pour seul objectif d'obtenir des informations sur l'adversaire. Si cela augmente la probabilité d'échecs, les agents discrets pourraient arriver à une situation où une négociation n'est pas profitable pour eux en moyenne. Dans ce cas, ils n'auraient plus aucune raison de négocier. La section suivante s'intéresse à un protocole qui évite ce genre de situation en obligeant les agents à révéler leur prix de réserve sans en informer l'adversaire. Ce protocole donne une incitation aux agents à ne négocier que dans le cas où ils peuvent trouver un accord, et à en rechercher véritablement un.

4.3 Protocole résistant aux agents curieux

Après avoir introduit les différents types d'agents et avoir formalisé le protocole dans lequel nous travaillons, nous nous proposons de modifier le protocole de *bargaining* afin de le rendre résistant au comportement des agents curieux. Nous commençons par décrire les raisons pour lesquelles le *bargaining* est vulnérable aux agents curieux, puis nous décrivons un protocole de négociation bilatérale basé sur le *bargaining*, et qui empêche les agents curieux de nuire. Nous en analysons les propriétés théoriques dans deux contextes : dans le cas où la négociation ne concerne qu'un prix, et dans le cas plus général de la négociation multi-attributs où les attributs sont continus. Nous démontrons deux théorèmes sur l'incitation à l'honnêteté de ce protocole, puis nous nous intéressons à l'effet de son introduction dans un contexte expérimental.

4.3.1 *Bargaining* et agents curieux

Le *bargaining* classique n'introduit aucun garde-fou pour empêcher un agent curieux de négocier juste pour collecter de l'information. C'est même un protocole particulièrement favorable aux agents curieux puisqu'ils sont avantagés dans chacune des phases.

Le domaine d'acceptabilité mutuel représente l'ensemble des propositions acceptables par l'acheteur et le vendeur tirées d'un historique.

Définition 4.7 (Domaine d'acceptabilité)
Pour un historique quelconque $h \in H$, on appelle domaine d'acceptabilité de i l'ensemble des propositions qui, une fois acceptées, génèrent une utilité positive pour l'agent i.

Soit u_i l'utilité de i. Pour un historique h, on appelle A_i le domaine d'acceptabilité de i :

$$A_i(h) = \{\pi \in \mathcal{D} | u_i(h; \pi; accept) \geq 0\} \tag{4.9}$$

Définition 4.8 (Domaine d'acceptabilité mutuel)
Pour un historique quelconque de $h \in H$, on appelle domaine d'acceptabilité mutuel $\Pi(h)$ l'intersection entre les domaines d'acceptabilité des agents :

$$\Pi(h) = A_1(h) \cap A_2(h) \tag{4.10}$$

Si le domaine d'acceptabilité mutuel est vide quel que soit l'historique, la négociation ne peut pas aboutir : chacun des agents préférera abandonner la négociation plutôt que d'accepter un compromis en dehors de cette zone d'acceptation. Dans ce cas, une négociation entamée implique que l'information révélée est inutile. Pour un agent curieux, une telle situation est avantageuse : elle aboutit à de l'information révélée sans rien payer en retour. Pour un agent discret, elle mènera nécessairement à une utilité négative (voir Théorèmes 4.1 et 4.2).

La séquence de propositions est définie par la stratégie et le prix de réserve de l'agent. L'adoption de stratégies faisant des concessions bien choisies peut augmenter le nombre d'échanges. Une telle stratégie peut être utilisée par un agent curieux afin d'optimiser l'information collectée.

L'issue de la négociation (accord ou rejet) est aussi un aspect primordial. Comme on l'a montré dans les Théorèmes 4.1 et 4.2, un agent curieux aura une utilité positive dès que des propositions sont faites même si la négociation s'achève par un rejet. *A contrario*, dans le cas d'un rejet, l'utilité d'un agent discret est négative. Cela met le rapport de force clairement en faveur d'un agent curieux, qui a la possibilité de n'accepter que si le prix proposé est nettement à son avantage.

Afin de réduire la vulnérabilité du *bargaining* aux agents curieux, nous proposons de concevoir un protocole qui réponde à chacune des failles que nous avons relevées. Nous proposons dans un premier temps un protocole qui se concentre sur la négociation du prix uniquement et ensuite un protocole qui s'intéresse au cas plus général de plusieurs attributs continus. Comme nous l'avons montré en Section 3.2, les techniques de modélisation d'adversaires sont avant tout des méthodes reposant sur l'apprentissage automatique. Or, en pratique, la quantité de données sera au moins aussi importante que la performance de l'algorithme les traitant, comme l'explique [Domingos, 2012], « En pratique, le moyen le plus rapide pour réussir est souvent juste d'obtenir plus de données. En règle générale, un algorithme naïf avec beaucoup de données en bat un plus intelligent avec de plus petites quantités de données » [2]. Nous proposons donc à partir de maintenant d'identifier l'information contenue par un historique à la longueur de ce dernier. L'utilité d'un agent dépend donc de l'issue de la négociation p et de la longueur de l'historique k. Par abus de notation, on étend la fonction d'utilité et on note $u_i(k, p)$ l'utilité d'un agent pour une négociation de k tours et se concluant sur un prix (ou un `reject`) p.

4.3.2 Cas mono-attribut : négociation sur le prix

De nombreuses négociations ont pour unique objet un prix. Dans ce cas, il est possible d'utiliser les caractéristiques particulières de cet attribut pour définir un protocole de négociation qui soit résistant aux agents curieux. Ces caractéristiques incluent notamment le fait qu'il s'agit d'un unique attribut continu, le fait que la fonction d'utilité des agents est monotone et que chacun sait dans quel

2. « *pragmatically the quickest path to success is often to just get more data. As a rule of thumb, a dumb algorithm with lots and lots of data beats a clever one with modest amounts of it.* »

sens varie cette fonction (croissante pour le vendeur, décroissante pour l'acheteur). Ces particularités permettent la définition d'un protocole simple et, comme nous l'expliquons dans la Section 4.4.2, une implémentation décentralisée.

4.3.2.1 Protocole

Afin d'éviter que les agents ne négocient pour obtenir de l'information plutôt que pour l'achat ou la vente d'un bien, nous définissons un nouveau protocole. Ce protocole s'appuie sur le protocole de *bargaining* standard, tel que défini dans la Section 2.2.2, mais aussi sur le mécanisme d'incitation de Myerson, introduit dans la Section 2.3.2.

Mise en correspondance Nous proposons d'abord d'éviter la perte d'information qui survient lorsque deux parties ayant des domaines d'acceptation incompatibles entrent en négociation. À cette fin, nous supposons l'existence d'un mécanisme permettant d'empêcher le *bargaining* lorsque aucun accord mutuel n'est possible. Les moyens de parvenir à une telle fin peuvent être centralisés ou non, et sont discutés en Section 4.4. Notons que le prix de réserve n'est pas unique lors de la négociation. Il est désormais une fonction de la longueur de l'historique, c'est à dire du nombre de propositions envoyées et reçues. Afin de nous assurer qu'un prix de réserve existe à tout moment de la négociation, les agents sont invités à donner leur pire prix de réserve.

Échange de propositions Nous proposons ensuite de borner le nombre d'échanges ; si aucun accord n'a été atteint avant que le nombre d'échange maximal ne soit atteint, le *bargaining* échoue. La borne associée à un *bargaining* b est notée k_b^{max}. Cette idée a été introduite par [Gneezy *et al.*, 2003] et soulève un problème : elle génère une asymétrie entre vendeur et acheteur. En effet, puisqu'il s'agit d'une borne, l'agent faisant la dernière proposition peut faire une très mauvaise proposition sous la forme d'une offre « à prendre ou à laisser ».

Cette problématique a été traitée par [Zhang et Liu, 2016]. Dans cet article, un médiateur reçoit les propositions des deux parties et les transmet une fois qu'il les a toutes les deux reçues. L'article traite la négociation dans le cas où il n'y a que le prix à négocier. Si les prix des agents se croisent (le prix proposé par le vendeur est supérieur à celui proposé par l'acheteur), l'échange est considéré comme un *accept* est la transaction est réalisée à la moyenne des prix proposés.

Bien que [Zhang et Liu, 2016] propose la présence d'un médiateur, cela n'est pas nécessaire en pratique. Cette question est également traitée dans la Section 4.4.

Obligation d'effectuer la transaction Finalement, nous proposons de prévenir les échecs possibles de la négociation en forçant l'acceptation de la transaction lorsqu'une proposition a été rejetée ou la borne atteinte. Dans une telle transaction forcée, le bien est échangé au prix de réserve des agents. L'argent restant est retiré de la transaction. Dans une place de marché, il peut par exemple être redistribué à l'ensemble des acteurs. Comme le montre le Théorème 4.5, cette situation est moins intéressante pour les agents que de trouver un terrain d'entente, et ils ont donc une forte incitation à en trouver un.

4.3.2.2 Analyse de l'incitation

Notre protocole a pour objectif de contrer les comportements des agents curieux, et de les forcer à se comporter comme des agents incurieux. En particulier, au début de la négociation, un agent curieux pourrait vouloir déclarer un prix qu'il ne voudrait pas payer, pour pouvoir entamer la négociation avec un autre agent et ainsi obtenir des informations de sa part. Nous démontrons dans cette section deux propriétés sur les incitations apportées par nos propositions :

- le théorème 4.3 montre que notre protocole donne une forte incitation pour un acheteur (resp. un vendeur) curieux à ne pas déclarer un prix trop haut (resp. trop bas). Pour que notre protocole incite les agents à être honnête, il faudrait qu'il incite également un acheteur (resp. un vendeur) curieux à ne pas déclarer un prix trop bas (resp. trop haut). Notre protocole n'a pas cette propriété, son incitation à l'honnêteté est donc partielle. La problématique est détaillée dans cette section après les Théorèmes 4.3 et 4.5,
- le théorème 4.5 montre que nos propositions donnent aussi une incitation à trouver un accord à la négociation.

Théorème 4.3 (Incitation partielle à l'honnêteté dans le cas mono-attribut)
Un protocole suivant les trois extensions décrites précédemment donne une incitation à un acheteur (resp. vendeur) curieux à déclarer un prix de réserve inférieur (reps. supérieur) à son véritable prix de réserve $\pi_b^1(k_b^{max})$ *(resp.* $\pi_b^2(k_b^{max})$*).*

Démonstration. Nous donnons ici la démonstration pour l'acheteur. La démonstration est identique pour le vendeur, *mutatis mutandis*. Pour simplifier les notations, on note $\pi_1 = \pi_b^1(k_b^{max})$, le prix de réserve de l'acheteur. On note $\hat{\pi_1}$, $\hat{\pi_2}$ les prix de réserve déclarés respectivement par l'acheteur et le vendeur. On note π le résultat de la négociation, et $u_1(n, \pi)$ l'utilité de l'acheteur en fonction de l'issue de la négociation et du nombre de propositions reçue. Pour démontrer ce théorème nous commençons par démontrer le lemme 4.4 puis nous montrons que la propriété est vérifiée dans tous les cas possibles.

Lemme 4.4
Si on identifie la quantité d'information reçue au nombre de propositions reçues, le prix de réserve d'un acheteur curieux est croissant en fonction du tour de la négociation.

Démonstration. Soient k_1, k_2, deux tours consécutifs de la négociation, et n_1, n_2 le nombre respectif de propositions reçues à ces tours.

Si k_1 **est pair** alors, le message suivant est envoyé par l'acheteur. Or $\pi_1(k) = \pi : u(\pi, n) = 0$. Ici, on a $n_1 = n_2$, et $u(n_1, \pi_1(k_1)) = 0 = u(n_2, \pi_1(k_2))$. Comme $n_1 = n_2$, on a $u(n_1, \pi_1(k_1)) = 0 = u(n_1, \pi_1(k_2))$. Supposons que $\pi_1(k_2) > \pi_1(k_1)$. Alors, on a $u(n_1, \pi_1(k_2)) < u(n_1, \pi_1(k_1))$, ce qui est contradictoire avec $u(n_1, \pi_1(k_2)) = u(n_1, \pi_1(k_1))$

Si k_1 **est impair** alors, le message d'après est envoyé par le vendeur. Si ce n'est pas une proposition, la démonstration est similaire au cas 1. Si c'est une proposition, alors, on a $n_2 > n_1$. Supposons $\pi_1(k_1) > \pi_1(k_2)$. Alors, on a

$$u_1(n_1, \pi_1(k_1)) < u_1(n_1, \pi_1(k_2)) \tag{4.11}$$

(par définition de l'acheteur). De plus,

$$u_1(n_2, \pi_1(k_2)) > u_1(n_1, \pi_1(k_2)) \tag{4.12}$$

par définition de l'agent curieux. Or, on a aussi

$$u_1(n_2, \pi_1(k_2)) = 0 = u_1(n_1, \pi_1(k_1)) \tag{4.13}$$

par définition du prix de réserve. Donc, en combinant les Équations (4.11) et (4.12), on obtient : $0 = u_1(n_2, \pi_1(k_2)) > u_1(n_1, \pi_1(k_2)) > u_1(n_1, \pi_1(k_1)) = 0$, ce qui est contradictoire. Donc, $\pi_1(k_1) \leq \pi_1(k_2)$

En combinant les deux cas, on obtient dans tous les cas : $\pi_1(k_1) \leq \pi_1(k_2)$, *Q.E.D.* □

Pour la suite de cette démonstration, nous comparons le cas où l'agent déclare son vrai prix de réserve, et celui où il déclare un prix de réserve $\hat{\pi_1} > \pi_1$. Nous procédons par disjonction des cas en faisant la comparaison pour chacun d'eux. Dans le tableau ci-dessous, on note u_{π_1} l'utilité obtenue si l'agent déclare π_1, et $u_{\hat{\pi_1}}$ l'utilité obtenue si l'acheteur déclare $\hat{\pi}$. Puisque $\hat{\pi_1} > \pi_1$, on peut distinguer 3 cas :

Cas	L'agent déclare $\hat{\pi_1} > \pi_1$	L'agent déclare π_1
$\hat{\pi_2} > \hat{\pi_1} > \pi_1$	La négociation ne commence pas : $u_{\hat{\pi_1}} = 0$	La négociation ne commence pas : $u_{\pi_1} = 0$
$\hat{\pi_1} \geq \hat{\pi_2} > \pi_1$	Le vendeur est rationnel et donc, s'il accepte une proposition au prix π, ce prix est meilleur que le prix de réserve qu'il a déclaré, donc $\pi \geq \hat{\pi_2}$	La négociation ne commence pas : $u_{\pi_1} = 0$
	Si la négociation termine avec le prix π, et au tour $k < k_b^{max}$ on a : $u_{\hat{\pi_1}} = u_1(k, \pi)$.	
	Si la négociation ne se termine pas (soit qu'un agent rejette une proposition soit que la borne est atteinte), on note k le tour auquel a eu lieu le `reject`, ou $k = k_b^{max}$ si la borne a été atteinte. Dans tous les cas, on obtient : $u_{\hat{\pi_1}} = u(k, \hat{\pi_1})$.	
$\hat{\pi_1} > \pi_1 \geq \hat{\pi_2}$	Les deux agents sont rationnels donc si la négociation se termine, le prix π vérifie $\hat{\pi_1} \geq \pi \geq \hat{\pi_2}$	Les deux agents sont rationnels donc si la négociation se termine, le prix π vérifie $\pi_1 \geq \pi \geq \hat{\pi_2}$
	Si la négociation se termine au tour k par un accord, $u_{\hat{\pi_1}} = u_1(k, \pi)$.	Si la négociation se termine au tour k par un accord, $u_{\pi_1} = u_1(k, \pi)$.
	Si la négociation échoue au tour k, $u_{\hat{\pi_1}} = u_1(k, \hat{\pi_1})$.	Si la négociation échoue au tour k, $u_{\pi_1} = u_1(k, \pi_1)$.

Comparons maintenant l'utilité obtenue pas les agents dans les différents cas :

$\hat{\pi_2} > \hat{\pi_1} > \pi_1$ L'utilité de l'agent ne change pas et est nulle.

$\hat{\pi_1} \geq \hat{\pi_2} > \pi_1$ On a $u_{\pi_1} = 0$. Si la négociation réussit, $u_{\hat{\pi_1}} = u_1(k, \pi)$, avec $\pi > \hat{\pi_2} > \pi_1$. De plus, $k \leq k_b^{max}$. Puisque u_1 est décroissante en fonction de π et croissante en fonction de k, on a $u_1(k, \pi) < u_1(k_b^{max}, \pi_1) = 0$. Donc, $u_{\hat{\pi_1}} < u_{\pi_1}$. Si la négociation échoue, on a $u_{\hat{\pi_1}} = u_1(k, \hat{\pi_1})$, or $k \leq k_b^{max}$ et $\hat{\pi_1} > \hat{\pi_2}$. Donc, $u_1(k, \hat{\pi_1}) < u_1(k_b^{max}, \pi_1) = 0$ et $u_{\hat{\pi_1}} < u_{\pi_1}$.

$\hat{\pi_1} > \pi_1 \geq \hat{\pi_2}$ Si la négociation réussit et que l'agent a déclaré $\hat{\pi_1}$, si $\pi \leq \pi_1$, alors, la même stratégie peut être utilisée pour obtenir le même résultat en déclarant π_1. Dans ce cas, $u_{\pi_1} = u_{\hat{\pi_1}}$. Si $\pi \geq \pi_1$, alors, dans le cas où l'agent a déclaré π_1, il peut rejeter la proposition et il obtient alors $u_{\pi_1} = u(k, \pi_1) \geq u(k, \pi) = u_{\hat{\pi_1}}$. Enfin, si la négociation échoue, on a $u_{\hat{\pi_1}} = u_1(k, \hat{\pi_1})$ et $u_{\pi_1} = u_1(k, \pi_1)$. Or $\pi_1 < \hat{\pi_1}$, donc $u_{\pi_1} > u_{\hat{\pi_1}}$.

Dans tous les cas, on obtient $u_{\pi_1} \geq u_{\hat{\pi_1}}$. L'agent n'a donc pas intérêt à déclarer un prix supérieur à son véritable prix de réserve. *Q.E.D* □

Théorème 4.5 (Incitation à rechercher un accord)
Si la négociation a commencé, rejeter une offre ou atteindre la borne est moins intéressant pour un couple d'agents négociant que trouver un quelconque accord meilleur que le prix de réserve qu'ils ont déclaré, avant que la négociation ne s'achève.

Démonstration. Soit une négociation s'interrompant au tour k, soit par un `reject`, soit parce que la borne a été atteinte. Soit $\hat{\pi_1}$ le prix déclaré par l'agent 1 et $\hat{\pi_2}$ le prix déclaré par l'agent 2. Comme la négociation a commencé, on a $\hat{\pi_1} > \hat{\pi_2}$. Donc, $\exists \pi \in \mathbb{R} : \hat{\pi_1} > \pi > \hat{\pi_2}$. Or, selon les définitions du vendeur et de l'acheteur,

$$\forall \pi \in \mathbb{R} : \hat{\pi_1} > \pi > \hat{\pi_2}, \begin{cases} u_1(k, \pi) > u_1(k, \hat{\pi_1}) \\ u_2(k, \pi) > u_2(k, \hat{\pi_2}) \end{cases} \tag{4.14}$$

Il est donc avantageux de remplacer la dernière proposition des agents par un accord au prix π. *Q.E.D.* □

Incitation à l'honnêteté Le protocole décrit tel quel n'incite pas à l'honnêteté. Supposons la situation suivante : un acheteur commence la négociation avec un prix de réserve de 20. S'il déclare son vrai prix de réserve, alors, il paiera au plus 20. S'il déclare plus, cela ne l'avantagera pas, et il risque de payer plus cher pour son bien. Mais il peut déclarer moins. Dans ce cas, il prend le risque que la négociation ne démarre pas parce que les prix de réserve ne sont pas compatibles. D'un autre côté, s'il déclare 10 au lieu de 20, il est sûr de ne pas payer plus de 10. Si c'est un agent incurieux et que sa fonction d'utilité est $u_1(\pi) = 20 - \pi$, alors, son utilité au pire est de 10 si la négociation commence. En fonction de sa propension à prendre des risques, cela peut être avantageux.

Ce problème peut sembler très proche de celui traité par Myerson. Cependant, la nature du mécanisme diffère : dans notre mécanisme, il y a un nombre infini de types possibles pour l'agent et l'utilité qu'il obtient ne dépend pas entièrement du mécanisme. Il est cependant possible d'utiliser le même principe que celui utilisé par Myerson : rendre la négociation non Pareto-optimale pour donner une incitation aux agents de ne pas déclarer un prix qui soit plus avantageux que leur vrai prix de réserve. Pour ce faire, il est possible de définir une probabilité que la négociation commence. Ainsi, pour un couple de prix déclarés $\hat{\pi_1}$, $\hat{\pi_2}$, on définit la probabilité de laisser la négociation démarrer par : $f\left(\max\left(\frac{\hat{\pi_1} - \hat{\pi_2}}{d}, 0\right)\right)$ où d est la taille du domaine de négociation (typiquement la plage de prix)

et $f : [0,1] \to [0,1]$ est une fonction croissante telle que $f(0) = 0$ et $f(1) = 1$. Si les agents ont des dispositions au risque quelconque, ce protocole n'est pas incitatif.

Donnons un exemple pour la fonction identité $f = \mathrm{id}_{[0,1]}$. Supposons une négociation sur un domaine constitué d'un unique attribut (un prix) variant sur $[0, 200]$. Supposons que le vendeur soit honnête : $\hat{\pi_2} = \pi_2(k_b^{max}) = 80$, et un acheteur dont le prix de réserve est $\pi_1(k_b^{max}) = 100$. La probabilité que la négociation commence est alors de 10%. Si l'acheteur déclare un prix de réserve $\hat{\pi_1} = 95$, la probabilité que la négociation démarre chute à 7.5%. Mais déclarer ce prix de réserve lui permettra possiblement de faire chuter le prix de 5. Tout dépendra alors de la négociation. Supposons que l'acheteur ait la fonction d'utilité incurieuse suivante : $u_1(\pi) = 100 - \pi$. Son utilité moyenne au pire passe de 0 à $7.5\% \times 5 = 0.375$. Son utilité au mieux passe, elle de 2 à 1.5. Tout dépend donc du résultat de la négociation.

Cette variante comporte en outre le même inconvénient que le mécanisme proposé par Myerson que nous avons décrit dans la Section 2.3.2. Il empêche parfois la négociation de commencer alors qu'elle aurait pu aboutir.

4.3.3 Cas multi-attributs

Nous nous sommes intéressés à définir un protocole dans le cas où la négociation ne porte pas que sur un prix, mais dans de nombreux cas, il est nécessaire de négocier sur plusieurs attributs. En effet, même dans une négociation triviale, comme sur un marché, il n'est pas rare de négocier le prix et la quantité de biens achetés. Nous nous intéressons donc à l'extension de notre protocole au cas multi-attributs.

4.3.3.1 Protocole

Voyons maintenant dans quelle mesure notre protocole peut être étendu au cas multi-attributs. Nous supposons que les attributs sont continus. Le protocole doit être modifié puisqu'il n'y a pas de prix de réserve dans le cas général. On raisonne alors non sur un prix de réserve mais sur le domaine acceptable de chaque agent i. Il s'agit de l'ensemble des propositions π telles que $\exists k \leq k_b^{max} : u_i(k, \pi) > 0$. On suppose en plus que cet ensemble est convexe et fermé, et que les fonctions d'utilité sont continues et *séparables*. Dans le cadre de cette thèse, on donne la définition suivante de la séparabilité pour une fonction d'utilité :

Définition 4.9 (Fonction d'utilité séparable)
Soit

$$u : \mathbb{N} \times \mathbb{R}^n \to \mathbb{R} \tag{4.15}$$
$$(k, \pi) \mapsto u(k, \pi) \tag{4.16}$$

u est dite séparable si et seulement si :

$$\begin{cases} \forall (k,k') \in \mathbb{N}^2, \forall (\pi,\pi') \in (\mathbb{R}^n)^2, u(k,\pi) > u(k,\pi') \implies u(k',\pi) > u(k',\pi') \\ \forall (k,k') \in \mathbb{N}^2, \forall (\pi,\pi') \in (\mathbb{R}^n)^2, u(k,\pi) > u(k',\pi) \implies u(k,\pi') > u(k',\pi') \end{cases} \tag{4.17}$$

On suppose enfin qu'il est possible que le résultat de la négociation soit différent pour l'acheteur et le vendeur dans le cas où ils ne trouvent pas d'accord. Cette supposition est forte, et dans le cas

général, elle n'est pas forcément remplie. On fait cette supposition dans le cas mono-attribut où seul le prix est négocié. Dans le cas où aucun accord n'est trouvé, en échange du produit, l'agent 1 paie plus et l'agent 2 reçoit moins. Dans le cas multi-attribut, cela ne va pas de soi. Prenons un exemple. Supposons que les agents souhaitent acheter un nouvel ordinateur, et que le budget de cet ordinateur est fixé. La négociation portera sur la manière de répartir ce budget entre les composants de l'ordinateur. Dans ce cas, il est impossible que l'issue de la négociation soit différente pour les agents. Notre supposition ne tient donc pas, et notre protocole ne peut être utilisé. Dans d'autres cas cependant, par exemple si l'un des attributs est un prix, notre supposition tient, et ce protocole est applicable. On utilise la même astuce que dans le cas mono attribut en faisant payer plus à un agent que l'autre ne reçoit et le surplus est perdu pour les agents.

On note $\mathcal{F}(A)$ la frontière topologique de A. Une définition formelle en est donnée dans [Mendelson, 1990]. La frontière d'un ensemble est son « bord ». Pour un ensemble d'acceptabilité, il s'agit de propositions qui sont acceptables, mais qui « touchent » des propositions qui ne le sont pas. Pour un domaine D, la frontière d'un ensemble d'acceptabilité de 1 à un tour k est représenté sur la Figure 4.3. Elles correspondent donc à des points pour lesquels l'utilité est nulle. Le protocole est identique, mais repose sur ces ensembles.

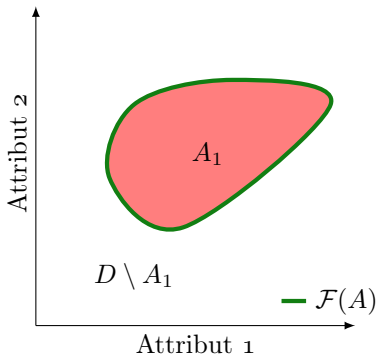


FIGURE 4.3 – Frontière d'un ensemble

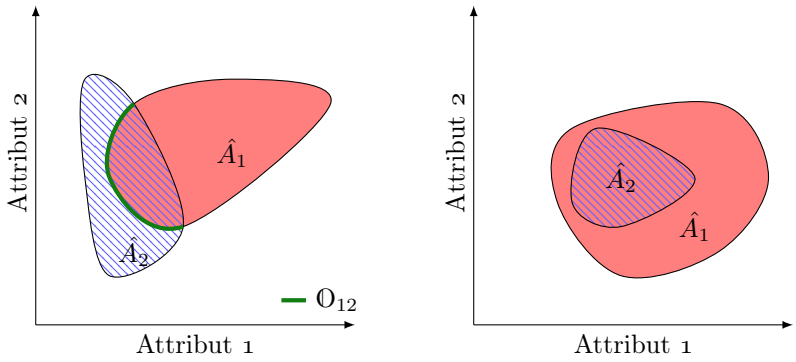
Mise en correspondance Les agents fournissent chacun leur domaine d'acceptabilité tel que donné en Définition 4.7. On note ces domaines respectivement A_1, A_2 et les domaines déclarés par les agents (potentiellement différents si les agents ont menti) $\hat{A}_1, \hat{A}_2$. La négociation commence si et seulement si $\hat{A}_1 \cup \hat{A}_2 \neq \varnothing$, c'est à dire s'il existe un ensemble de propositions qui fournissent une utilité positive aux agents.

Échange de propositions Le protocole reste le même à une exception près. On suppose que le médiateur connaît les domaines d'acceptabilité des agents. Il lui est donc impossible de déterminer si les utilités des agents se croisent. Ainsi, dans cette variante, les agents doivent accepter la proposition de l'adversaire pour qu'il y ait un accord. Dans le cas où un `accept` et un `reject` sont reçu, c'est l'`accept` qui prime. Dans le cas où deux `accept` sont reçus, une des deux propositions précédentes est choisie aléatoirement.

Obligation d'effectuer la transaction Il est nécessaire de définir l'accord imposé aux agents dans le cas où la négociation échoue. S'il existe un point du domaine de négociation π sur la frontière de

$\hat{A}_1$ et sur celle de $\hat{A}_2$, $\mathcal{F}(\hat{A}_1) \cap \mathcal{F}(\hat{A}_2)$ (telle que décrite dans la Figure 4.3) il peut être tentant de choisir ce point comme solution à la négociation. Les deux agents reçoivent alors, comme dans le cas mono-attribut, une utilité nulle en cas de désaccord, ce qui les pousse à trouver un accord. Cette solution est mathématiquement élégante, mais elle permet à chaque agent de connaître un point de la frontière de l'ensemble d'acceptabilité de l'adversaire, *i.e.* un point auquel son utilité est nulle, ce qui est une information précieuse. Pour pallier ce problème, nous ajoutons l'hypothèse qu'il est possible que l'issue de la négociation soit différente pour l'acheteur et le vendeur.

La transaction forcée est donc plus complexe dans le cas multi-attribut que dans le cas mono attribut. Pour deux ensembles acceptables $\hat{A}_1, \hat{A}_2$, soit $\mathbb{O}_{12} = \mathcal{F}(\hat{A}_1) \cap \hat{A}_2$ Cet ensemble représente les proposition qui sont à la frontière pour 1 (dont l'utilité est nulle) et qui sont acceptables pour 2. On peut alors distinguer deux cas : $\mathbb{O}_{12} \neq \varnothing$ ou $\mathbb{O}_{12} = \varnothing$. Ces deux cas sont représentés dans la Figure 4.4, pour un domaine de négociation à 2 attributs. Notons que si $\mathbb{O}_{12} = \varnothing$, on a forcément $\hat{A}_2 \subsetneq \hat{A}_1$, puisque $\hat{A}_2 \cap \hat{A}_1 \neq \varnothing$ grâce à l'étape de mise en relation.



(a) $\mathbb{O}_{12} \neq \varnothing$ (b) $\mathbb{O}_{12} = \varnothing$

FIGURE 4.4 – Existence de $\mathbb{O}_{12}$

Dans le premier cas, on choisit un élément de $\mathbb{O}_{12}$ de manière aléatoire comme solution pour l'agent 1. Dans le second cas, un élément aléatoire de $\mathcal{F}(\hat{A}_1)$ est choisi. Comme nous l'avons précisé, on restreint ce choix de manière à ce uqe 1 et 2 puissent avoir des issues différentes à la négociation.

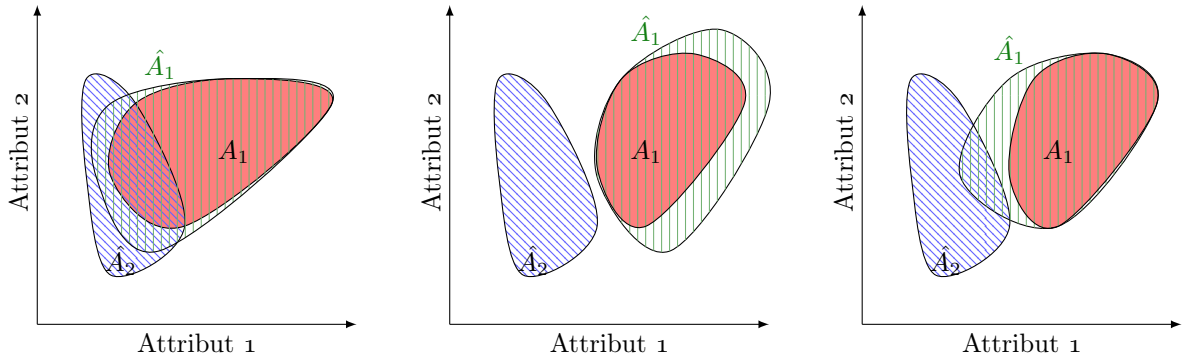
4.3.3.2 Analyse de l'incitation

Vérifions maintenant que les Théorèmes 4.3 et 4.5 sont toujours valables dans le cas multi attributs :

Théorème 4.6 (Incitation partielle à l'honnêteté dans le cas multi-attributs)
Soit A_i l'ensemble d'acceptabilité de l'agent i, c'est à dire l'ensemble des propositions pouvant amener une utilité positive à un certain tour de la négociation, et $\hat{A}_i$ l'ensemble d'acceptabilité déclaré. Alors, un agent curieux n'a pas intérêt à déclarer un $\hat{A}_i$ tel que $A_i \subsetneq \hat{A}_i$

Démonstration. Sans perte de généralité, nous démontrons la propriété pour l'agent 1. Nous procédons par disjonction sur $A = A_1 \cap \hat{A}_2$ (de la même manière qu'on raisonnait sur π_1 et $\hat{\pi}_2$ dans le cas mono-attribut). Les cas sont représentés sur la Figure 4.5.

La démonstration de chaque cas est très semblable à celle du Théorème 4.3. Notons en outre que les agents sont curieux, ce qui signifie que leur domaine d'acceptabilité s'agrandit au fur et à mesure



(a) Cas 1 (b) Cas 2.1 (c) Cas 2.2

FIGURE 4.5 – Les trois cas possibles de la démonstration du Théorème 4.6

de la négociation (de la même façon que leur prix de réserve augmente dans le cas mono attribut). Notons également que puisque par définition de A_1

$$\forall x \in A_1 \setminus \hat{A}_1, \forall x' \in A_1, \exists h \in H : u_1(x,h) \geq 0, u_1(x',h) < 0 \tag{4.18}$$

. Étant donné qu'on a supposé les fonctions d'utilité séparables au sens de la Définition 4.9, $\forall h \in H, u_1(x,h) \geq u_1(x',h)$.

Cas 1 : $A_1 \cap \hat{A}_2 \neq \varnothing$. Dans ce cas, la négociation démarre de toute façon. Si la négociation réussit, et que la solution retenue est dans A_1, alors, les deux solutions sont identiques et les utilités aussi. Si la solution retenue est dans $\hat{A}_1 \setminus A_1$, alors, l'utilité de l'agent est pire qu'un `reject` dans le cas où l'agent a déclaré A_1, puisqu'un reject lui attribut un point de A_1. Dans le cas où la négociation n'aboutit pas, si l'agent déclare $\hat{A}_1$:

- soit il reçoit une utilité dans $\hat{A}_1 \setminus A_1$ et son utilité est pire que s'il avait déclaré A_1
- soit il reçoit une utilité dans A_1 et son utilité est identique qu'il déclare A_1 ou $\hat{A}_1$.

Cas 2 : $A_1 \cap \hat{A}_2 = \varnothing$. Dans ce cas, si $\hat{A}_1 \cap \hat{A}_2 = \varnothing$ (cas 2.1), quel que soit le domaine déclaré par l'agent, il reçoit une utilité nulle (la négociation ne démarre pas). Si $\hat{A}_1 \cap \hat{A}_2 \neq \varnothing$ (cas 2.2), soit la négociation se termine par un accord dans $\hat{A}_1 \setminus A_1$ (puisque l'agent 2 est rationnel) et l'agent 1 reçoit une utilité négative, soit la négociation ne termine pas, et l'agent reçoit une utilité dans $\mathbb{O}_{12}$, or $\mathbb{O}_{12} \subset \hat{A}_2$ et $A_1 \cap \hat{A}_2 = \varnothing$, donc $\mathbb{O}_{12} \cap \hat{A}_2 = \varnothing$, et donc l'agent reçoit une utilité négative, au lieu de 0 dans le cas où l'agent déclare A_1. □

La démonstration du Théorème 4.5 peut être reprise telle quelle en remplaçant les prix de réserve par des ensembles d'acceptation.

En faisant un certain nombre de suppositions sur les fonctions d'utilité des agents et sur le domaine de négociation, il est possible de transposer notre protocole au cas multilatéral. Les suppositions faites ne sont cependant pas toujours vérifiées : dans le cas où la négociation dépend d'un prix et où tous les attributs sont continus, il est raisonnable de penser que ce protocole est utilisable. Pour des négociations sortant de ce cadre, cela dépendra avant tout du domaine de la négociation et de la possibilité d'attribuer un résultat différent à l'acheteur et au vendeur.

4.3.4 Discussion

Notre modification du *bargaining* nous a permis de limiter l'influence néfaste des agents curieux. D'un point de vue théorique, le protocole que nous proposons a des propriétés intéressantes : encouragement à trouver un accord, et incitation partielle à l'honnêteté – une incitation à ne pas déclarer un domaine d'acceptabilité plus grand que celui de l'agent. L'incitation totale à l'honnêteté n'a pu être obtenue, et la variante permettant de limiter le gain à déclarer domaine d'acceptabilité plus restreint que le domaine de l'agent a pour conséquence de réduire le nombre de négociations qui démarrent, même lorsque trouver un accord serait possible, sans donner aucune garantie théorique. Pour ces raisons, nous nous limitons au protocole défini par trois modifications :

- la déclaration du domaine d'acceptabilité,
- la limitation du nombre de propositions échangées et
- l'obligation de payer le prix de réserve (ou d'obtenir l'utilité de réserve) dans le cas où les agents n'arrivent pas à un accord.

4.4 Implémentation

Dans la section précédente, nous avons laissé de côté un aspect capital de notre protocole : son implémentation. Si certaines modifications ne concernent que le protocole lui-même et ne nécessitent pas qu'on prenne en compte leur implémentation concrète, un aspect fondamental de notre protocole est la déclaration du prix de réserve. Cette information est totalement privée, et il est impossible que les agents déclarent directement leur prix de réserve à leur adversaire pour savoir si le *bargaining* doit ou non commencer. Il est donc nécessaire que cette information soit échangée sans que les agents puissent apprendre le prix effectif de l'adversaire. Nous proposons à cette fin deux protocoles. Le premier est centralisé et utilise un médiateur. Ce médiateur est un tiers de confiance, et peut être considéré comme les médiateurs de [Myerson et Satterthwaite, 1983, Osborne et Rubinstein, 1990].

Cependant, cette solution n'est pas pleinement satisfaisante. Comme l'explique [Sandholm, 1999], « Toutes choses égales par ailleurs, les protocoles distribués devraient être préférés entre autres afin d'éviter les points de défaillances uniques et les goulots d'étranglement » [3]. Nous proposons donc une implémentation décentralisée basée sur une résolution du problème des deux millionnaires [4], et qui a été introduit par [Yao, 1982]. La solution que nous utilisons, et qui est issue de [Ioannidis et Grama, 2003], est notamment utilisée pour des problématiques de cryptographie (voir notamment [Yao, 1986, Chaum *et al.*, 1988]).

Enfin, nous analysons l'application de ce protocole à une place de marché, dans un contexte où de multiples acheteurs et de multiples vendeurs interagissent. Pour ce faire, nous introduisons un protocole hybride basé sur les enchères de Vickrey, et qui conserve les propriétés de notre protocole.

4.4.1 Implémentation centralisée

La première manière de mettre en correspondance les agents est d'utiliser un tiers de confiance. Cette solution est déjà utilisée dans les mécanismes que nous avons décrits dans la Section 2.3.2

3. « *All else being equal, distributed protocols should be preferred to avoid a single point of failure and a performance bottleneck* »

4. *Two millionaire's problem*

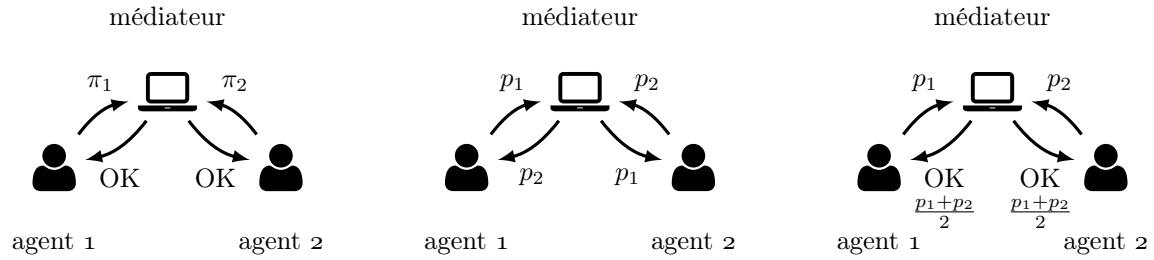
[Myerson et Satterthwaite, 1983, Nash, 1950] ainsi que par d'autres travaux portant sur la négociation automatique [Osborne et Rubinstein, 1990]. La différence fondamentale entre ces travaux et notre proposition est le rôle que joue notre médiateur.

4.4.1.1 Mise en relation

Fondamentalement, le médiateur est là pour collecter les informations privées des agents et les exploiter. Ces informations sont importantes pour trouver un compromis qui satisfasse les deux agents, mais elle sont trop sensibles pour être envoyées directement à l'adversaire. Afin d'exploiter ces informations malgré tout, le médiateur collecte les domaines d'acceptabilité des agents et lance la négociation s'il existe un domaine d'acceptabilité mutuel.

4.4.1.2 Échange de propositions

Durant l'échange de propositions, le médiateur est chargé de recevoir les offres des deux agents et de les leur transmettre simultanément. La Figure 4.6 illustre l'implémentation centralisée. Le médiateur met les agents en relation dans la Figure 4.6a, il leur transmet les messages de manière simultanée, comme décrit dans la Figure 4.6b et il les informe quand les prix se croisent et qu'un accord a donc été trouvé dans la Figure 4.6c.



(a) Mise en relation (b) Échange de propositions (c) Accord

FIGURE 4.6 – Implémentation centralisée du protocole résistant aux agents curieux

4.4.2 Implémentation décentralisée

La solution centralisée du problème demande la présence d'un médiateur, qui constitue une source potentielle de panne et demande à ce que les agents aient confiance dans ce tiers. Il est plus rassurant pour les agents d'utiliser une solution décentralisée. Cette solution consiste avant tout à envoyer des informations sans que celles-ci puissent être lues par l'adversaire. Cela est possible dans le cas où les agents négocient uniquement un prix. Il y a deux questions à se poser : d'abord, les agents doivent être capables de savoir si la négociation doit commencer ou non. Ensuite, ils doivent pouvoir envoyer leurs propositions tout en empêchant l'adversaire de les voir tant que lui-même n'a pas envoyé la sienne.

4.4.2.1 Mise en relation

Dans ce cas, savoir s'il existe un domaine d'accord mutuel revient à savoir si le prix de réserve de l'acheteur est supérieur à celui du vendeur. Il faut cependant que cette information soit transmise sans qu'il soit possible à l'autre agent de déduire le prix de réserve lui-même. C'est précisément le problème dit « des deux millionnaires » introduit par [Yao, 1982]. Ce problème concerne deux agents qui souhaitent savoir lequel est le plus riche des deux sans révéler leur fortune à qui que ce soit. Ce problème a été résolu par [Yao, 1982, Lin et Tzeng, 2005, Ioannidis et Grama, 2003]. Nous donnons ici la solution proposée par [Yao, 1982] sous forme de tableau dans l'Algorithme 2. Les prix de réserve des agent 1 et 2 sont respectivement représentés par m et n. On note M une borne sup du prix de réserve. L'algorithme fonctionne comme suit :

1. L'agent 1 génère une paire clé publique/clé privée grâce au protocole RSA.
2. L'agent 2 génère une valeur aléatoire k connue de lui seul, qu'il chiffre à l'aide de la clé de l'agent 1. Il caclcule l en ajoutant son propre prix de réserve n. Ce calcul dépend de la valeur de k. L'agent 1 ne peut donc pas déduire n de ce nombre.
3. L'agent 1 calcule (v_i). Il faut noter que

$$v_n = D_1(E_1(k) - n + n) = D_1(E_1(k)) = k \tag{4.19}$$

   et

$$\forall j \neq n, v_j = D_1(E_1(k) - n + j) \tag{4.20}$$

4. L'agent 1 ajoute l'information sur sont propre prix de réserve. Pour ce faire, il choisit un nombre premier $2 < p \leq \frac{M}{2}$. Ce nombre sert à calculer les $v_i \mod p$. Il ajoute ensuite 1 si $i \geq m$, 0 sinon. Puisque $v_n = k$, on a

$$z_n = \begin{cases} k \mod p \text{ si n<m} \\ k \mod p+1 \text{ sinon} \end{cases} \tag{4.21}$$

5. L'agent 2 fait ce calcul et renvoie le résultat à l'agent 1.

Nous ne reproduisons pas les démonstrations de la correction du résultat ni de l'impossibilité pour chacun des agent de déduire le prix de réserve de l'adversaire de ce protocole. Nous laissons aussi de côté les conditions précises de création de différentes variables de cet algorithme, qui se trouvent dans la publication originale de [Yao, 1982]. Cet algorithme demande un grand nombre de calculs pour la production de (z_i), et cette information transite par le réseau. D'autres protocoles reposant sur des notions de mathématiques plus avancées permettent également de résoudre ce problème, comme l'ont notamment montré [Ioannidis et Grama, 2003, Lin et Tzeng, 2005].

Échange de propositions Dans l'implémentation centralisée, le médiateur tient deux rôles : il décide ou non de lancer la négociation et il gère l'échange de propositions. Pour cette seconde tâche, il attend de recevoir les propositions des deux agents avant de les transmettre à leurs adversaires respectifs.

Il y a cependant une autre manière, décentralisée, de traiter cette tâche. Pour ce faire, il est possible de s'appuyer sur des algorithmes de chiffrement à clé symétrique. Nous ne décrirons pas ces algorithmes ici, car cela est hors du cadre de cette thèse. Le lecteur pourra se référer à [Agrawal et Mishra, 2012]. Nous nous contenterons donc de décrire leur fonctionnement général. Dans un chiffrement à clé symétrique, un agent émetteur, appelé ici agent 1 cherche à envoyer un message de manière sécurisée à un agent 2. L'agent 1 génère une clé (une suite de bits) qui est ensuite utilisée dans un algorithme permettant d'encoder le message à envoyer. La clé et le message sont envoyées séparément, de préférence par des canaux différents. L'agent 2 peut utiliser la clé reçue pour déchiffrer le message.

Notre implémentation décentralisée de l'échange d'offres est décrite en Figure 4.7. Le protocole procède comme suit : les agents produisent une clé et envoient leur proposition chiffrée à l'aide de la clé. Quand un agent reçoit le message chiffré, il peut envoyer la clé pour que son adversaire puisse lire le message. Une clé est générée pour chaque message.

Ces deux procédures décentralisées constituent une implémentation où il est possible de se passer de médiateur central, ce qui évite un point potentiel de panne. Il est cependant nécessaire que le domaine

Algorithme 2 Protocole des deux millionnaires

	Agent 1				**Agent 2**		
	calcul	secret	public		public	secret	calcul
(1)	Crée (E_1, D_1) une paire de clés RSA	m, D_1	E_1	$\longrightarrow$	E_1	n	
(2)			l	$\longleftarrow$	l	k, x	Génère un nombre aléatoire k ; calcule $x = E_1(k); l = x - n$
(3)	Calcule $(v_i)_{i \leq M}$ $v_i = D_1(l - i)$	(v_i)					
(4)	Choisit p premier ; calcule $(z_i)_{i \leq M}$ $z_i = v_i \mod p + \mathbb{1}_{i \geq m}$		$(z_i), p$	$\longrightarrow$	z_i		
(5)							Calcule $z_n \mod p$. Si $z_n \mod p = k \mod p$ alors, $n > m$, sinon $n \leq m$.

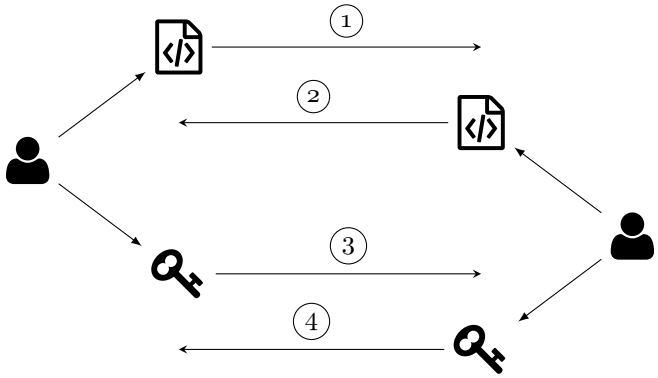


FIGURE 4.7 – Implémentation décentralisée de l'échange de messages

de négociation se résume à un prix pour que la première fonctionne. En effet, s'il est possible de chiffrer n'importe quel message, le protocole de Yao, décrit ci-dessus, sert à comparer deux nombres. Dans le cas où il faut comparer des domaines d'utilité, cela devient impossible, et la présence d'un tiers capable d'une telle analyse est nécessaire.

4.4.3 Implémentation 1:n

Nous avons vu deux manières d'implémenter notre protocole résistant aux agents curieux dans le cas où nous avons deux agents qui négocient l'un avec l'autre. Comme nous l'avons indiqué dans le Chapitre 1, dans le contexte de notre plate-forme, un protocole permettant de gérer la présence de plusieurs acheteurs potentiels est souhaitable. Nous présentons donc un protocole 1:n basé sur les enchères de Vickrey (que nous avons décrites dans la Section 2.2.1.1), qui permet de conserver les incitations de notre protocole.

Pour un bien, les acheteurs intéressés soumettent secrètement leur prix de réserve sur ce bien. Les prix au-dessous du prix de réserve du vendeur sont supprimés. S'il reste au moins deux prix de réserve, celui qui a fait l'offre la plus haute est sélectionné. C'est là que notre protocole diffère des enchères de Vickrey : une fois le gagnant sélectionné, ce dernier ne paiera pas le prix annoncé mais pourra négocier selon notre protocole résistant aux agents curieux avec le vendeur, avec le prix de réserve qu'il a déclaré. Sa première offre sera cependant nécessairement le prix de réserve de la seconde meilleure offre lors de l'étape d'enchères. S'il y a moins de deux acheteurs au-dessus du prix de réserve du vendeur, la négociation est annulée.

Ce protocole permet de généraliser notre protocole au cas 1:n tout en conservant ses propriétés. En effet, le Théorème 4.5 est toujours valable, et comme nous le démontrons, le Théorème 4.3 aussi.

Théorème 4.7 (Incitation partielle à l'honnêteté de l'implémentation 1:n)
Dans le protocole de Vickrey-bargaining, le Théorème 4.3 reste valide.

Démonstration. La démonstration est identique à celle de l'incitation de l'enchère de Vickrey, présenté dans la Théorème 2.1. Supposons que l'agent fasse une offre au-dessus de son évaluation.

S'il est seul à enchérir, il reçoit une utilité nulle dans tous les cas. Sinon

- soit la seconde meilleure offre était plus haute que son prix de réserve, et dans ce cas, il commence à négocier au-dessus de son prix de réserve. Dans tous les cas, que la négociation réussisse ou non, il devra donc payer un prix au-dessus de son prix de réserve, et obtiendra une utilité négative.
- soit la seconde meilleure offre est au-dessous de son prix de réserve, et il aurait remporté l'enchère de toute façon, et avec un prix de réserve (appliqué si les agents ne trouvent pas d'accord) plus faible (et donc meilleur pour lui).

□

L'illustration de ce théorème est donné en Figure 4.8. Si un agent (ici représenté en rouge) essaie d'augmenter son prix de réserve et que cela le fait gagner, il passera par une négociation, qui engendrera nécessairement une utilité négative. À droite, les deux acheteurs ayant le prix le plus bas sont éliminés parce que leur prix de réserve est inférieur à celui de l'acheteur.

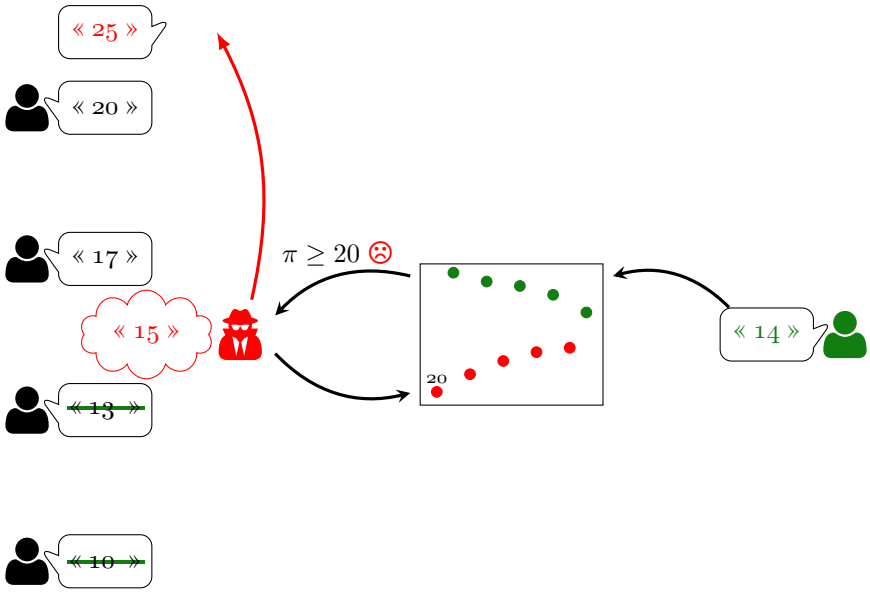


FIGURE 4.8 – Illustration de l'implémentation 1:n

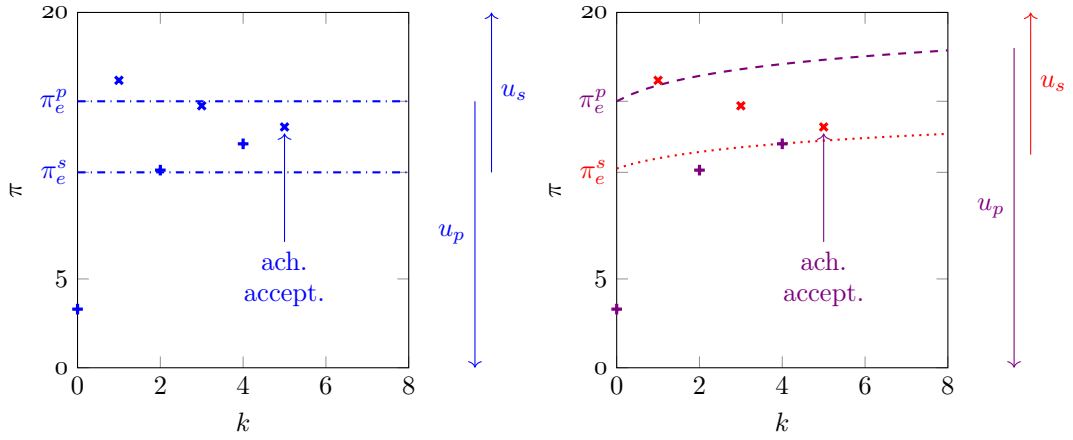
L'implémentation que nous avons conçue pour le contexte 1:n conserve les propriétés de notre protocole. Bien qu'il ne soit pas incitatif, le fait que les agents soient mis en concurrence encourage les acheteurs à ne pas baisser leur prix de réserve, afin d'éviter de voir d'autres agents passer devant eux. Ce protocole conserve en outre la possibilité pour les agents de négocier comme dans un *bargaining* standard.

4.4.4 Analyse expérimentale

Dans cette partie, nous proposons une analyse empirique sur l'impact de notre protocole. Les expérimentations sont menées dans le cas où le domaine de négociation se résume à un prix. Nous proposons en outre d'identifier quel impact a chaque modification apporte au *bargaining*. Nous décrivons d'abord le protocole expérimental. Nous calibrons ensuite la borne de négociation que nous utilisons. Enfin, nous présentons le bien-être des agents.

4.4.4.1 Protocole expérimental

Nos expérimentations consistent à lancer plusieurs instances de *bargaining* mono-attribut. Les instances de *bargaining* considérée sont définies par l'inclusion ou non d'une ou plusieurs modifications que nous avons introduites en Section 4.3.2. Ces variantes sont le *bargaining* classique, le *bargaining* avec mise en correspondance, avec une borne donnée ou avec les trois extensions ensemble (les deux précédentes et obligation de transaction). L'obligation de transaction n'a de sens que si les agents se sont engagés sur leur prix de réserve, puisque la transaction se fait à ce prix. Nous évaluons donc cette modification avec les deux autres déjà implémentées.



(a) Négociation entre deux agents incurieux

(b) Stratégie et prix de réserve d'un acheteur curieux et d'un vendeur discret

π^e cur. π^e inc. π^e diss. strategie acheteur strategie vendeur

FIGURE 4.9 – Deux exemples de négociations et prix de réserve pour le *bargaining* mono-attribut (prix)

Un agent a est associé à un type (*incurieux*, *curieux* ou *discret*), à un prix de réserve initial $\pi_a(0)$, à une fonction d'utilité u_a, et à une fonction de stratégie que nous definissons ci-dessous. Les fonctions d'utilité des agents sont [5] :

- pour les agents incurieux :

$$\forall b' \in \mathcal{B}, u_a(b) = \begin{cases} \pi_1 - \pi_b & \text{si succès} \\ 0 & \text{sinon} \end{cases} \tag{4.22}$$

- pour les agents curieux :

$$\forall b' \in \mathcal{B}, u_a(b) = \begin{cases} \pi_1(0) \cdot \dfrac{1}{\log_n(n+k)} - \pi_b & \text{si succès} \\ \pi_1(0) \cdot \left(\dfrac{1}{\log_n(n+k)} - 1 \right) & \text{sinon} \end{cases} \tag{4.23}$$

5. Seule l'utilité de l'acheteur est donnée ici, l'utilité du vendeur étant similaire *mutatis mutandis*.

- pour les agents discrets :

$$\forall b' \in \mathcal{B}, u_a(b) = \begin{cases} \pi_1(0) \cdot \log_n(n+k) - \pi_b & \text{si succès} \\ \pi_1(0) \cdot (\log_n(n+k) - 1) & \text{sinon} \end{cases} \tag{4.24}$$

Ces équations correspondent à l'utilité affichée dans la Figure 4.9. La stratégie d'offres est une adaptation de la fonction fournie par [Faratin *et al.*, 1998] et discutée dans la Section 3.3.1 :

$$\gamma + \left(\kappa_a + (1 - \kappa_a) \cdot \left(\frac{\nu}{\nu_a^{max}} \right)^{\frac{1}{\beta_a}} \right) \cdot (\gamma_a - \pi_a^b) \tag{4.25}$$

Elle renvoie à chaque étape la proposition que l'agent fera. La stratégie d'acceptation est une stratégie simple définie dans la Section 3.4.1 : si l'autre partie fait une meilleure proposition que celle que l'agent aurait faite, l'agent l'accepte ; si la proposition calculée est pire que le prix de réserve de l'agent, il l'abandonne. Les paramètres κ_a, β_a et γ_a déterminent la forme de la courbe pour l'agent a. Des exemples sont tracés dans la Figure 4.9.

Une expérience est définie par un couple d'agents, définis par leur type, et une instance de *bargaining*. Nous observons le bien-être des agents. Chaque résultat présenté a été moyenné sur 10.000 instances de marchandage où les prix de réserve des agents sont issus d'une distribution gaussienne et κ_a, β_a et γ_a sont tirés selon une distribution uniforme. Nous implémentons le protocole centralisé pour le cas mono-attribut du prix, tel que défini dans la Section 4.4.1. Nous séparons chacune des modifications que nous avons apportées au protocole afin d'étudier l'impact de chacune, à l'exception de la dernière qui repose sur les autres, et qui est donc évalué en même temps qu'elles.

4.4.4.2 Calibration de la borne

Notre seconde modification consiste à introduire une borne au-delà de laquelle la négociation s'interrompt, provoquant soit une annulation de la transaction, soit dans le second cas une transaction forcée au prix de réserve. Il convient cependant de choisir cette valeur avec attention. Elle doit permettre aux agents de négocier, sans quoi aucune négociation n'aboutira. Elle doit aussi interrompre les négociations trop longues qui sont le fait d'un allongement artificiel induit par les agents curieux.

La Figure 4.10 présente la variation du bien-être [6] en fonction de la valeur de la borne. Chaque graphique se focalise sur un type d'agent particulier et rend compte de l'évolution de son bien-être lorsqu'il est face à chaque autre type d'agent.

Nous observons que toutes les courbes se composent de deux parties. Quand l'échéance est inférieure à certaines valeurs (500 pour les agents discrets, 750 pour les incurieux et 1000 pour les curieux), elle a un fort impact sur le marchandage. Jusqu'à cette borne, un nombre significatif de négociations sont interrompues du fait de l'échéance. Au-delà, la variation de l'échéance a souvent moins d'impact : elle ne limite plus que très peu de négociations. L'objectif de l'introduction de cette borne est d'éviter de rallonger les négociations plus que nécessaire. Cependant, il ne faut pas non plus que cette borne empêche les agents de négocier. En particulier, il faut éviter que cette borne interrompe les négociations

6. Dans le cadre de cette thèse, nous utilisons le bien-être utilitaire, *i.e.* la moyenne de l'utilité des agents

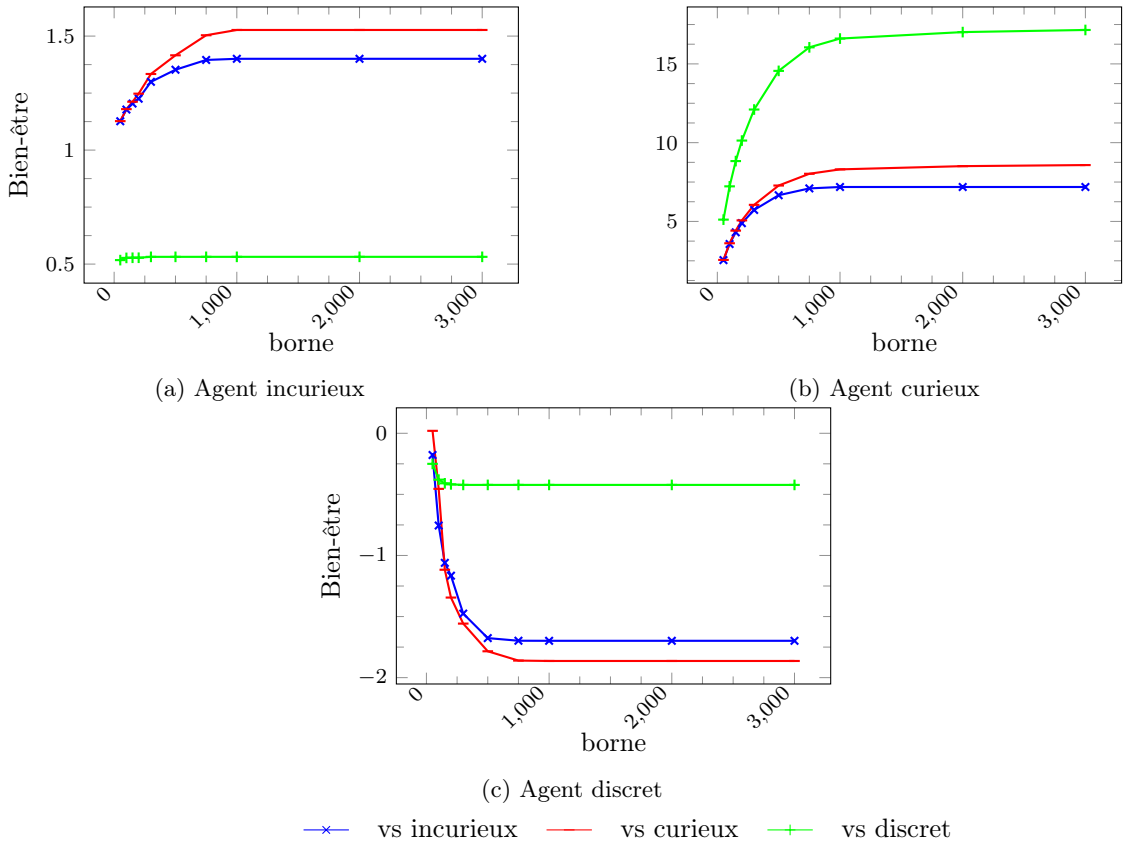


FIGURE 4.10 – Impact de la borne sur le bien-être des différents types d'agents

des agents discrets. Pour ce faire, nous choisissons un k_b^{max} pour lequel le bien-être des agents discrets ne changent plus. Cela signifie que les négociations dans lesquelles les agents discrets sont impliqués ne sont plus interrompues par cette borne, et donc que la borne interrompra désormais les négociations qui durent plus longtemps, et induisent une perte d'information pour les agents discrets. La valeur optimale est 500. Nous l'utiliserons donc pour les expériences ultérieures limitant le nombre de tours.

4.4.4.3 Impact sur le bien-être

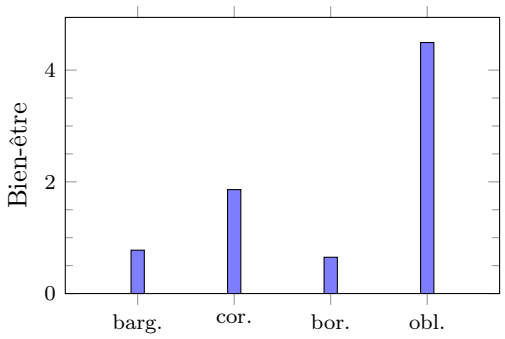
La Figure 4.11 montre le bien-être des différents types d'agents dans les différentes situations. « barg » est utilisé pour le marchandage standard (*bargaining*), « cor. » pour la mise en correspondance, « bor. » pour la borne, avec une borne à 500 messages et « obl. » pour toutes les extensions à la fois. La légende de chaque sous-figure indique les types d'agents qui négocient et quel bien-être est représenté.

La Figure 4.11a montre comment les extensions affectent le bien-être des agents classiques. Seule la seconde réduit un peu leur bien-être (10%), puisque borner le nombre de messages conduit les plus longues négociations à se terminer par des rejets. La correspondance le double en augmentant le nombre de négociations réussies. Appliquer toutes les extensions quadruple leur bien-être, puisque les agents adaptent leur stratégie afin d'atteindre leur prix de réserve avant l'échéance (*cf.* Théorème 4.5) : certains agents qui prévoyaient d'étendre la négociation dans le marchandage standard font maintenant

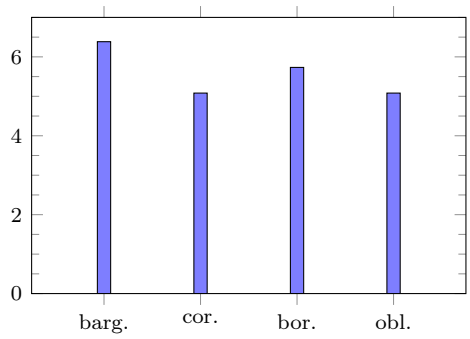
des concessions plus rapidement.

La Figure 4.11b montre comment les agents curieux sont impactés par les extensions que nous proposons dans le contexte classique. Tous les principes diminuent leur bien-être. La correspondance le diminue d'un sixième : les agents arrivent à un accord plus souvent et plus rapidement, parfois à un prix très élevé. La borne diminue le bien-être de 10%, l'agent curieux obtenant moins d'information. Appliquer tous les principes diminue également le bien-être des agents curieux d'un sixième.

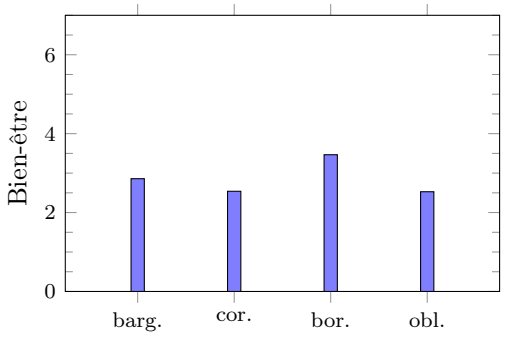
La Figure 4.11d montre l'impact des extensions que nous proposons sur les agents discrets face à des agents incurieux. Ici aussi, chaque extension a un impact positif. Le bien-être moyen des agents



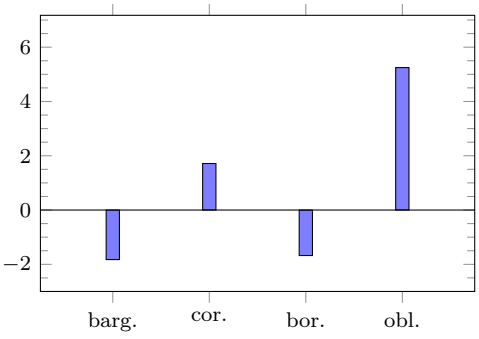
(a) agents incurieux face à des incurieux



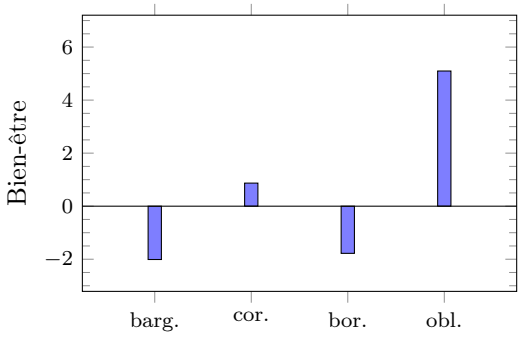
(b) agents curieux face à des incurieux



(c) agents curieux face à des discrets



(d) agents discrets face à des incurieux



(e) agents discrets face à des curieux

FIGURE 4.11 – Bien-être des agents, dans différentes situations

discrets est négatif sur le marchandage classique parce que beaucoup de négociations s'achèvent par un échec, comme nous l'avons expliqué plus tôt. La correspondance limite considérablement le nombre de rejets, et résulte en un bien-être positif pour les agents discrets avec la même valeur absolue. Ajouter une borne n'augmente que peu leur bien-être (il reste négatif, sa valeur absolue est diminuée de 9%). Cela réduit en effet le nombre de messages échangés avant un rejet. Finalement, forcer la transaction augmente beaucoup leur bien-être pour la même raison que pour les agents incurieux.

Les Figures 4.11c et 4.11e décrivent comment la balance des forces est impactée par nos extensions quand des agents curieux rencontrent des agents discrets. Les fig-sec2-wlf,fig-sec-wlf montrent à peu près la même valeur. On peut noter cependant que le bien-être des discrets est toujours moindre lorsqu'ils négocient avec des curieux. De même, Figures 4.11b et 4.11c montrent les mêmes évolutions, le bien-être des agents curieux est pire lorsqu'ils négocient avec les discrets plutôt qu'avec les curieux. La seule exception est le cas de la borne, qui amène un bien-être plus haut pour les agents curieux : plus de négociations sont rejetées, et ils obtiennent donc plus d'informations « gratuitement ».

4.4.5 Discussion

Comme nous l'avons vu il est possible dans le cas où la négociation porte sur un prix d'utiliser une implémentation décentralisée. Lorsque la négociation porte sur plusieurs attributs et même si le protocole est applicable (si le domaine est continu, les domaines d'acceptation convexes et les fonctions d'utilité continues), il est impossible d'utiliser cette implémentation. Il est alors nécessaire de recourir à un tiers de confiance, avec toutes les limitations de trafic et de sécurité que cela implique.

Dans le cas où ce protocole est appliqué à une plate-forme, il est souhaitable d'avoir un protocole multilatéral, au moins de type 1:n. Pour ce cas, nous avons donné un protocole inspiré des enchères de Vickrey. Après une étape d'enchère où les agents révèlent secrètement leur prix de réserve au commissaire, les agents sont invités à négocier selon notre protocole bilatéral, avec pour prix de départ le second meilleur prix de réserve déclaré. Ce protocole conserve les propriétés du protocole bilatéral, mais n'est pas un mécanisme de Groves. En effet, le prix de réserve de l'acheteur reste celui qu'il a déclaré. L'agent subit cependant la pression des autres agents, et risque de passer de la première à la seconde position s'il déclare un prix de réserve trop faible. Cela l'incite à ne pas déclarer un prix de réserve trop bas.

D'un point de vue expérimental, notre implémentation d'agents curieux et discrets nous a permis de tester notre protocole et de constater son efficacité. Il améliore considérablement l'utilité des agents discrets tout en réduisant celle des agents curieux.

4.5 Conclusion

À notre connaissance, la curiosité en négociation n'a jamais été étudié dans ce contexte. Avec l'essor de la négociation automatique, l'augmentation de la puissance de calcul et le développement de nouvelles techniques d'apprentissage, cette problématique doit être traitée. Cela est d'autant plus vrai dans un contexte où les agents ne disposent pas des mêmes moyens.

Au-delà de la définition que nous avons donnée, nous proposons également un protocole qui incite les agents à ne pas être curieux. Ce protocole est composé de trois étapes : une étape de mise en relation, où les agents sont mis en relation si et seulement si leurs prix de réserve sont compatibles,

une étape de *bargaining* classique et une résolution de la négociation en fonction de si les agents ont trouvé ou non un accord. Du fait de l'utilisation du prix de réserve des agents dans le pire des cas, notre protocole ne peut être un protocole de Groves. Il donne cependant aux agents une incitation à ne pas mentir pour négocier avec des agents afin d'obtenir de l'information sans payer. De plus, nous avons montré comment il est possible de pousser les agents à agir honnêtement. Bien que notre protocole ne soit pas incitatif, ces modifications peuvent permettre, dans la pratique, de pousser les agents à indiquer des prix proches de leur véritables prix de réserve.

Nous avons proposé une extension de ce protocole au cas multilatéral, en restreignant le domaine de négociation et les fonctions d'utilité possibles des agents. Nous avons montré que dans ce contexte, les agents ont toujours les mêmes incitations en terme d'honnêteté. Nous avons enfin proposé deux implémentations du protocole bilatéral et une extension à un protocole basé sur les enchères. Nous avons fourni une implémentation décentralisée pour le cas où les agents négocient uniquement un prix, et une implémentation centralisée qui fonctionne dans tous les cas.

Nous espérons que ce cette problématique sera prise en compte dans d'autres places de marché multi-agents. Notre définition est assez générale pour être utilisée dans d'autres travaux, et permettre une extension aux domaines discrets. Une autre perspective est la définition que nous avons donnée du *bargaining*, et qui s'appuie sur les jeux extensifs nous paraît plus pertinente pour analyser ce protocole. Elle permet de faire le pont entre le domaine de la négociation automatique et la théorie des jeux d'une part, mais aussi entre la négociation automatique et les techniques d'intelligence artificielle utilisées pour la conception de joueurs, et qui ont récemment connu de nombreux succès.

Chapitre 5

MoCaNA : Un agent de négociation basé sur MCTS

“
Our new worldwide robot economy may develop its own problems, and for that reason we have the Machines. The Earth's economy is stable, and will remain stable, because it is based upon the decisions of calculating machines that have the good of humanity at heart through the overwhelming force of the First Law of Robotics.
Isaac ASIMOV, *I, Robot*, 1950
”

Sommaire

5.1 Introduction 124
5.2 Problématique 125
5.2.1 Méthodes de Monte Carlo 126
5.2.2 Problématiques spécifiques à la négociation 130
5.2.3 Discussion 131
5.3 Un *Monte Carlo-based Negotiation Agent* (MoCaNA) 132
5.3.1 Modélisation d'adversaire 133
5.3.2 Une stratégie d'offre basée sur MCTS 144
5.3.3 Stratégie d'acceptation 146
5.3.4 Discussion 146
5.4 Analyse expérimentale 147
5.4.1 Détails d'implémentation 147
5.4.2 Protocole expérimental 149
5.4.3 Résultats 150
5.4.4 Discussion 153
5.5 Conclusion 154

5.1 Introduction

Imaginez-vous, en plus de vos tâches habituelles, devoir passer plusieurs heures régulièrement à négocier pour obtenir un prêt. C'est ce que vivent au quotidien les PME. Ce temps passé à négocier est perdu pour faire d'autres choses : produire, trouver de nouveaux clients ou de nouveaux prospects. Si la décision est prise d'embaucher quelqu'un qui se dédie à cette tâche, c'est une personne de moins à travailler autre part. D'autant que financer le BFR n'est pas chose aisée, en particulier pour une petite entreprise. Même dans le cadre d'une plate-forme ouverte telle que la nôtre, où le prêt s'adosse au client de l'entreprise, la plate-forme est ouverte et les vendeurs sont donc en concurrence les uns avec les autres. La négociation d'un prix par lot de factures notamment demande une attention toute particulière, et se révèle donc particulièrement chronophage. Pour accepter de financer un lot, certains investisseurs peuvent également demander que d'autres aspects des lots de factures fassent également partie de la négociation ; il peut par exemple s'agir de remplacer des factures en fonction de leur client ou de leur date de remboursement.

Ce manque à gagner peut cependant être réduit au moyen d'une intelligence artificielle. En effet, la négociation automatique, que nous avons présentée dans le Chapitre 3 est un domaine en plein essor. Les agents de négociation automatique présentent de nombreux avantages. Ils sont capables de modéliser l'autre partie afin de la « comprendre » et d'échanger des milliers de propositions sans que l'utilisateur n'ait à intervenir. Enfin, ils sont adaptatifs et sont donc capables de négocier efficacement avec un nouvel interlocuteur.

Après avoir présenté un protocole de négociation pour la plate-forme, nous nous focalisons maintenant sur la conception d'un agent de négociation automatique. Nous nous intéressons en particulier à une famille de méthodes qui a reçu beaucoup d'attention dans le domaine de l'IA pour les jeux : les méthodes de Monte-Carlo. Ces méthodes ont plusieurs avantages : elles sont adaptatives et efficaces, et ont récemment occupé le devant de la scène suite à la victoire d'AlphaGo contre les plus grands joueurs de Go mondiaux. AlphaGo, décrit par [Silver *et al.*, 2016] utilise de manière conjointe les méthodes de Monte-Carlo en interaction avec des réseaux de neurones profonds. L'une des difficultés principales du Go est son grand facteur de branchement (le nombre de coups possible à chaque instant dans le jeu). Contrairement aux échecs dont le facteur de branchement initial est de l'ordre de la dizaine, le facteur de branchement initial du Go est de 361. Cela rend impossible la construction d'un arbre de type minimax assez profond pour donner de bons résultats. Comme on l'a vu, la négociation automatique peut être vue comme un jeu extensif, dont le facteur de branchement correspond à l'ensemble des propositions possibles. Ce nombre peut être très élevé, et même être infini dans le cas où le domaine contient des attributs continus. On peut alors être surpris que les méthodes de Monte-Carlo n'aient pas été utilisées dans ce cadre.

Un autre avantage des méthodes de Monte-Carlo est qu'elles sont très adaptatives. Elles constituent d'ailleurs l'une des méthodes les plus utilisées dans le *General Game Playing*, comme le montrent [Méhat et Cazenave, 2008, Finnsson, 2012]. Le *General Game Playing* est un domaine qui consiste à fournir à un agent un ensemble de règles inconnues auparavant dans un formalisme particulier et lui demander de jouer. Bien que le protocole que nous avons présenté dispose d'une borne, ce protocole n'est pas parfait. En particulier, il ne permet pas de gérer les attributs catégoriels. Il est donc possible que les acteurs préfèrent négocier dans un contexte plus général, notamment en utilisant le protocole de *bargaining* standard. Dans ce contexte, la borne n'est pas nécessairement présente. Dans ce chapitre nous mettons à l'épreuve les méthodes de Monte-Carlo afin de voir si elles sont capables de s'adapter à la négociation automatique dans ce contexte.

Du point de vue de la négociation, nous souhaitons concevoir un agent capable de négocier dans

un contexte très général. Nous souhaitons ainsi que notre agent puisse non seulement négocier dans un contexte identique à celui que fournit l'ANAC, mais aussi qu'il puisse négocier dans un contexte plus général. En particulier, nous souhaitons que notre agent soit capable de négocier dans des domaines de négociation composés de tous types d'attributs, qu'ils soient catégoriels, entiers ou continus. Nous souhaitons également que notre agent soit capable de négocier sans aucune connaissance qu'une quelconque borne, qu'il s'agisse d'une borne publique connue des agents ou d'une borne privée connue de l'adversaire seulement et qui devrait être inférée par modélisation de l'adversaire. Ces pré-requis correspondent aux cas d'application réels de notre plate-forme. D'une part, les négociations dans une situation réelle ne font aucune supposition sur les attributs du domaine. Il peut par exemple s'agir de négocier à la fois le taux de rabais d'une facture, sa source et le nombre de jours avant remboursement. D'autre part, dans de de nombreuses situations, le protocole n'indique pas de limite de temps ou de tour à la négociation et les agents ne négocient pas toujours avec un agenda en tête. Cela ne signifie pas qu'il continueront la négociation éternellement, mais ils peuvent tout à fait négocier sans avoir aucune idée du moment où ils s'arrêteront. Dans ce cas, il est impossible d'inférer le moment où l'adversaire coupera court à la négociation en se basant sur son comportement, puisque lui-même ne sait pas à quel moment cela arrivera.

À cette fin, nous proposons la conception d'un agent de négociation basé sur la recherche arborescente de Monte-Carlo (*Monte Carlo Tree Search* ou MCTS) ainsi que sur des méthodes de modélisation de l'adversaire. La complexité du domaine nous pousse à sortir des sentiers battus des implémentations les plus communes de MCTS basées sur UCT et à changer les stratégies que l'algorithme utilise habituellement.

Afin d'exposer les tenants et les aboutissants de l'application de MCTS à ce problème, nous présentons brièvement les algorithmes de Monte-Carlo ainsi que les particularités de la négociation qui imposent l'adaptation de ces algorithmes à notre contexte. Nous proposons ensuite une description de notre agent en utilisant le formalisme BOA, qui nous a servi de base lorsque nous avons fait une revue des agents de négociation automatique au Chapitre 3. Notre agent se compose d'un module de modélisation de la fonction d'utilité de l'adversaire et de sa stratégie, d'une stratégie d'offre basée sur MCTS et d'une stratégie d'acceptation. Nous présentons également deux variantes de notre stratégie d'offre, en mettant en place un élagage de l'arbre de jeu. La première variante consiste à déterminer un seuil d'utilité en deçà duquel les propositions ne sont pas prises en compte. Pour la seconde variante, nous utilisons les propositions de l'adversaire. Nous analysons cet agent face à un agent de type *Random Walker*, décrit par [Baarslag et Hindriks, 2013] qui n'utilise aucune connaissance sur la borne. Nous analysons ensuite les résultats et mettons notre agent aux finalistes de l'ANAC 2014. Notre agent se retrouve alors dans une situation compliquée : alors que les finalistes de l'ANAC appuient leur stratégie sur la connaissance de la borne et sur le fait que le domaine de négociation se compose d'entiers, compose d'entiers, notre agent fait abstraction de telles présuppositions. Nous concluons enfin en montrant quelles sont les pistes ouvertes par ce travail. En particulier, il s'agit à notre connaissance du premier pont entre les méthodes de Monte-Carlo et la négociation automatique.

5.2 Problématique

Appliquer les méthodes de Monte-Carlo à la négociation ne va pas de soi. En effet, la négociation est un jeu particulier, bien différent des échecs, du go ou même du skat. En particulier, le facteur de branchement de la négociation en tant que jeu ne change pas. À chaque tour, il est possible de faire n'importe quelle proposition du domaine de négociation. Ce domaine est lui-même particulier. Dans ce chapitre, nous ne faisons pas de présupposition sur sa forme. Il peut contenir des attributs entiers,

discrets et même continus. Les autres jeux que nous avons cités ne laissent pas cette possibilité ; il est impossible de jouer entre deux cases aux échecs.

Dans cette section, nous présentons les grandes lignes des méthodes de Monte Carlo. Nous expliquons ce qui a fait leur succès dans d'autres jeux et donnons les motivations qui nous ont poussé à les utiliser dans le cadre de la négociation automatique au sein de notre plate-forme. Nous énonçons ensuite les particularités de la négociation en tant que jeu tel que présentée dans la Définition 4.2, et en quoi ces particularités imposent de traiter la négociation différemment des jeux tels que les échecs ou le go.

5.2.1 Méthodes de Monte Carlo

La première publication sur les méthodes de Monte Carlo date de 1949 et a été écrite par [Metropolis et Ulam, 1949]. Dans cet article et les suivants, les méthodes de Monte Carlo ont été utilisées pour approximer des intégrales de fonctions dont le calcul était trop complexe pour être réalisé par les moyens habituels. Ces méthodes reposent sur la loi des grands nombres, et l'idée qu'en faisant assez de tirages aléatoires, la moyenne des résultats obtenus tendra vers le résultat attendu. Quand le calcul concerne l'intégrale d'une fonction, il s'agit de calculer la surface sous la courbe. On tire alors un certain nombre de points de manière aléatoire entre une borne supérieure de la fonction et une borne inférieure sur l'intervalle sur lequel on veut intégrer, puis on vérifie si les points sont au-dessous de la courbe. En utilisant la loi des grands nombres, la proportion de points au-dessous de la courbe tend vers la probabilité qu'un point soit au-dessous de la courbe. On obtient alors la proportion entre l'aire sous la courbe et la surface totale, ce qui permet de calculer l'intégrale. Cette méthode est décrite dans la Figure 5.1.

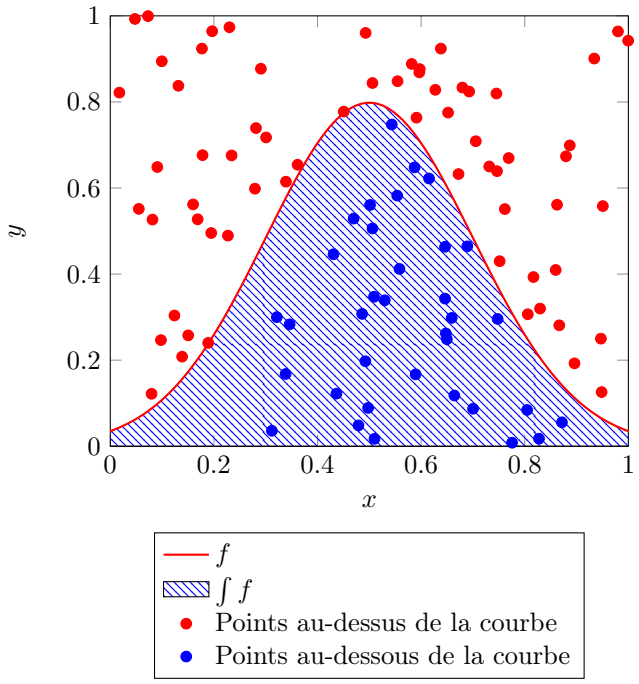


FIGURE 5.1 – Calcul d'intégrale par la méthode de Monte-Carlo

En intelligence artificielle pour les jeux, on cherche à approximer non plus la valeur d'une intégrale mais le score moyen obtenu à la fin du jeu si un coup est joué. À cette fin, on simule aléatoirement un jeu jusqu'à ce qu'un historique terminal soit constitué. On calcule alors le score des joueurs. Les méthodes de Monte-Carlo les plus simples dans ce contexte sont les méthodes de Monte Carlo « à plat » ou *flat Monte Carlo* en anglais, qui fonctionnent sur le même principe que les méthodes de Monte-Carlo utilisées pour approximer les intégrales. Cependant, les joueurs ne jouent pas de manière aléatoire, ce que ces méthodes ne permettent pas de modéliser. Des méthodes ont donc proposé de mêler les méthodes de Monte-Carlo aux méthodes d'exploration d'arbres de jeux. Cela a donné naissance à une nouvelle famille de méthodes, nommées recherches arborescentes de Monte Carlo, ou *Monte Carlo Tree Search*.

5.2.1.1 Monte-Carlo à plat

Dans les jeux à grand facteur de branchement, il peut être compliqué de construire un arbre minimax avec suffisamment de profondeur pour qu'il soit pertinent. D'autre part, certains jeux sont difficilement évaluables tant qu'un historique terminal n'a pas été atteint. C'est le cas du Go où il peut être difficile de déterminer quel joueur « est en train de gagner », contrairement aux échecs où de telles heuristiques, basées sur la valeur des pièces perdues et le degré de liberté de celles qui restent peuvent servir à déterminer quel joueur a le plus de chances de l'emporter.

Dans ce cas, les approches consistant à construire un arbre minimax sont impossibles, et il est nécessaire d'utiliser d'autres approches. Une possibilité est de s'appuyer sur des simulations de jeux menées jusqu'à leur terme et d'assigner à chaque coup le score moyen de ces simulations. La stratégie du joueur adverse étant inconnue, on simule habituellement le jeu de manière aléatoire. Bien que cette approche repose sur le résultats de simulations aléatoires et donc peu représentatifs d'une stratégie réaliste, elle s'est montrée très efficace notamment, comme le montre [Cazenave, 2006] pour les jeux à information imparfaite comme le Phantom Go, où elles restent encore les plus efficaces aujourd'hui.

Certaines améliorations ont été apportées à ce modèle. La première modification consiste à simuler plus souvent les mouvements qui ont été peu simulés et ceux qui ont eu de bons résultats, en s'appuyant sur un critère nommé UCB pour *Upper Confidence Bound*. Ce critère est à la base de l'*Upper Confidence Tree*, et nous le décrivons plus en détail dans la section suivante. Une autre modification consiste à imbriquer les simulations de Monte-Carlo : chaque mouvement de la simulation est généré en réalisant un Monte-Carlo à plat. Cette méthode nommée *Nested Monte Carlo* a notamment été utilisée pour les jeux à information imparfaite, comme l'expliquent [Furtak et Buro, 2013].

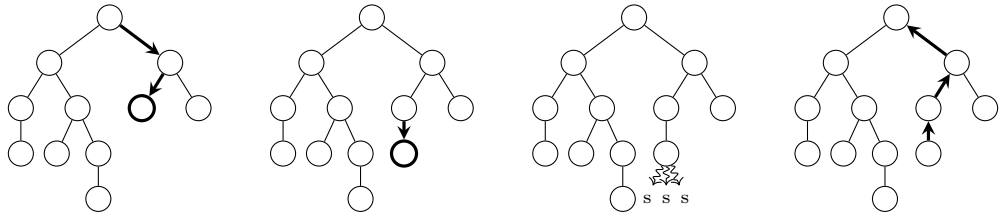
5.2.1.2 Recherche arborescente de Monte-Carlo

Il n'en reste pas moins que pour les autres jeux, cette supposition que les agents jouent de manière aléatoire n'est pas réaliste, et nuit à la pertinence des scores obtenus. Il est donc nécessaire d'adapter cette technique au contexte des jeux. [Kocsis et Szepesvári, 2006] proposent donc une méthode hybridant la construction d'un arbre de jeu, méthode qui a fait ses preuves, et les méthodes de Monte-Carlo. Chaque nœud de l'arbre de jeu garde en mémoire les scores obtenus lors des simulations où il a été joué, ainsi que le nombre de simulations faites dans la branche partant du nœud. Pendant la construction de cet arbre, les branches privilégiées sont celles qui ont été peu explorées et celles qui ont le plus haut score dans les simulations.

L'arbre ainsi généré est nommé UCT (*Upper Confidence Tree*). MCTS est un algorithme *anytime*,

Algorithme 3 MCTS

function MCTS(Historique h, StratégieSelection $sp1$, StratégieExpansion ep, StratégieSimulation $sp2$)
    Nœud $n \leftarrow$ Initialiser()
    Nœud *selectionne*
    Double *score*
    while Il y a du temps **do**
        $selectionne \leftarrow$ Sélection$(n, sp1)$
        $selectionne \leftarrow$ Expansion$(selectionne, ep)$
        $score \leftarrow$ Simulation$(selectionne, sp2)$
        Rétropropagation(*selectionne*, *score*)
    end while
    return $\text{argmax}_{coup} \{n.fils.score\}$
end function
function SÉLECTION(Nœud n, StratégieSimulation p)
    while ¬ExpansionNécessaire(n) **do** ▷ ExpansionNécessaire décide si n doit être étendu
        $n \leftarrow \arg\max_{fils \in n.fils}\{p(fils)\}$ ▷ p détermine si un fils doit être exploré
    end while
    return n
end function
function EXPANSION(Nœud n, StratégieExpansion p)
    Coup $m \leftarrow \arg\max_{\{coup|(n,coup)\in H\}} \{p(coup)\}$ ▷ Où H est défini comme dans Définition 4.2
    Historique $h \leftarrow (n, m)$
    Nœud n
    $f.pere \leftarrow n$
    $f.fils \leftarrow \varnothing$
    $f.racine \leftarrow h$
    $f.score \leftarrow 0$
    $f.visite \leftarrow 0$
    $f.fils \leftarrow n.fils \cup f$
    return f
end function
function SIMULATION(Nœud n, StratégieSimulation p)
    Historique $h \leftarrow n$
    while h n'est pas terminal **do**
        $h \leftarrow (h, p(h))$
    end while
    return Score(h)
end function
procedure RÉTROPROPAGATION(Nœud n, Double s)
    while $n \neq \varnothing$ **do**
        $n.score \leftarrow (n.score + s) \times \dfrac{n.visite}{n.visite + 1}$ ▷ Calculer la moyenne
        $n.visite \leftarrow n.visite + 1$
        $n \leftarrow n.pere$
    end while
end procedure



(a) Sélection (b) Expansion (c) Simulation (d) Rétropropagation

FIGURE 5.2 – MCTS tel que présenté par [Browne *et al.*, 2012]

ce qui signifie qu'il peut être interrompu à tout moment. Dans les faits, on laissera un certain temps à l'algorithme puis on lui demandera de donner le coup qu'il estime être le meilleur. Plus on simule chaque nœud fils de la racine, plus l'arbre est exploré en profondeur et plus le score de ces nœuds est représentatif – *i.e.* proche de son score minimax. Il est cependant nécessaire de pouvoir choisir entre plusieurs alternatives et donc d'explorer un certain nombre de nœuds fils de la racine, c'est à dire explorer l'arbre en largeur. Cette dichotomie s'appelle le dilemme exploration-exploitation. La stratégie de sélection de MCTS ainsi que le critère permettent de trancher. Lors du parcours de l'arbre, la stratégie de sélection détermine à la fois s'il y a lieu d'étendre un nouveau nœud et si ce n'est pas le cas, quel fils du nœud sélectionné sera le prochain à être sélectionné.

La fonction décrite dans l'Algorithme 3 représente ce fonctionnement. n représente le nœud sélectionné qui est mis à jour à chaque passage de la boucle. Le nœud sélectionné est celui qui maximise le critère de la stratégie de sélection. Le résultat du point de vue de l'arbre est représenté sur la Figure 5.2a. Lorsque le critère d'expansion, représenté par la fonction `ExpansionNécessaire` est rempli, un nouveau nœud fils du nœud sélectionné est étendu, comme le montre la Figure 5.2b. La politique d'expansion détermine comment il est créé. Il peut s'agir de le sélectionner aléatoirement parmi les coups possibles après le coup du nœud sélectionné, mais il est aussi possible de déterminer un sous-ensemble qui ne sera pas considéré, ou de favoriser certains mouvements au détriment d'autres. La manière de favoriser l'extension de certains coups au détriment d'autres constitue la stratégie d'expansion qui est mentionnée dans l'Algorithme 3. Quand un nœud est étendu, il sert de point de départ à une simulation. Chaque mouvement des agents est simulé en suivant une stratégie prédéfinie, souvent choisie pour avoir un coût calculatoire faible. Cette étape est représentée dans la Figure 5.2c. Quand un état final est atteint, l'agent doit avoir un moyen de déterminer le score des joueurs. Dans un jeu à information complète, l'agent peut directement déduire cette information de l'historique du jeu. Dans un jeu à information incomplète, d'autres méthodes sont à envisager. Il peut s'agir d'inférer l'utilité de l'adversaire ou de créer plusieurs arbres pour les différentes utilités possibles. Enfin, une fois ce score calculé, il est répercuté sur tous les nœuds de l'arbre qui ont été sélectionnés pendant la phase de sélection, avec un poids inversement proportionnel au nombre de fois que le nœud a déjà été sélectionné. Cette étape est représentée sur la figure Figure 5.2d.

Nous décrivons maintenant les stratégies utilisées pour la sélection, l'expansion et la simulation. Pour la sélection et l'expansion, la stratégie la plus utilisée est l'*Upper Confidence Bound* (UCB) [Kocsis et Szepesvári, 2006]. UCB étend un nouveau nœud suivant une stratégie d'expansion aléatoire tant que tous les nœuds fils n'ont pas encore été étendus. Si tous les nœuds fils ont déjà été étendu, UCB sélectionne le nœud j maximisant le critère C_j vérifiant

$$C_j = \overline{X_j} + \sqrt{\frac{2 \cdot \ln(n)}{n_j}} \tag{5.1}$$

où $\overline{X_j}$ est le score du nœud j, n_j est le nombre de simulations faites alors que le nœud j a été sélectionné, et n le nombre global de simulations. Cette stratégie a plusieurs avantages :

- D'abord, l'Équation (5.1) montre qu'UCB apporte une réponse au dilemme exploration/exploitation. La première partie du critère représente l'exploitation : le but est de sélectionner les nœuds ayant un score élevé. La seconde partie du critère concerne l'exploration : l'algrithme cherche aussi à sélectionner les nœuds qui ont été peu sélectionnés en regard du nombre total de simulations.
- Ensuite, le critère a une propriété intéressante, démontrée par [Kocsis et Szepesvári, 2006] : l'arbre tend vers l'arbre minimax lorsque le nombre de simulations tend vers l'infini.

Afin d'obtenir de bons résultats même avec peu de simulations, l'algorithme UCT a été modifié. Ces améliorations sont notamment présentées dans [Browne *et al.*, 2012]. On trouve notamment RAVE (pour *Rapid Action Value Evaluation*) et AMAF (pour *All Moves As First*). Ces modifications proposent de faire une somme pondérée des scores de tous les nœuds dont les mouvements sont identiques, indépendamment de l'historique de jeu. Cela permet de donner une évaluation pertinente des nœuds, même dans le cas où il n'est pas possible de faire beaucoup de simulations. D'autres améliorations proposent d'élaguer certaines branches en utilisant les connaissances sur le jeu. Cette connaissance peut aussi être utilisée pour ne pas explorer l'arbre jusqu'au bout quand certaines combinaisons ou des tables de fin de jeu connues sont atteintes.

Ces améliorations concernent soit l'algorithme MCTS lui-même, soit UCB. Cependant, cette stratégie ne peut être utilisé que dans le cas où le nombre de mouvement possibles est fini et « petit », de l'ordre du millier. Dans le cas où le domaine est continu, ou s'il y a un très grand nombre d'actions possibles, UCT ne peut être utilisé, en raison de sa politique d'expansion. Si le nombre d'expansions possible est infini, l'algorithme continue à étendre de nouveaux nœuds jusqu'à ce que le nombre limite de simulations soit atteint, et chaque mouvement n'est simulé qu'une unique fois. Si le facteur de branchement est trop important, les coups possibles ne sont simulés qu'un petit nombre de fois, et les scores sont donc très peu fiables car la loi des grands nombres ne se vérifie pas.

Pour de tels domaines, il est cependant possible d'utiliser d'autres stratégies. [Rolet *et al.*, 2009a, Rolet *et al.*, 2009b, Couëtoux, 2013] proposent une solution : le critère du *progressive widenining*. Il s'agit d'une fonction qui prend en entrée le nombre de fois où un nœud a été visité, et le nombre de ses fils qui ont été étendu. Cette méthode sert de base à la stratégie d'offres de notre agent. Elle est décrite en détail dans la Section 5.3.2.1.

Une autre problématique est que l'UCT est conçu pour les jeux à deux joueurs à somme nulle ; il converge d'ailleurs vers l'arbre minimax, un arbre de jeu utilisé pour ce type de jeux. Pour des jeux à somme non nulle ou à plus de deux joueurs, il est nécessaire d'utiliser un autre critère. En particulier, les scores de chaque nœud doivent être calculés pour chaque joueur, afin que ce critère soit utilisé par la suite dans la stratégie de sélection, à la manière d'une recherche $\max^n$[Sturtevant et Korf, 2000].

Enfin, la stratégie la plus générale dans l'étape de simulation est de choisir un coup légale de manière aléatoire. Cependant, comme le montrent [Ponsen *et al.*, 2010], il peut être intéressant d'utiliser un modèle d'adversaire lors des simulations lorsque cela est possible. Cela permet de rendre chaque simulation plus pertinente et donc d'obtenir un résultat plus conforme à la réalité.

5.2.2 Problématiques spécifiques à la négociation

Les jeux ne sont pas tous identiques. En particulier, la négociation est un jeu très différent de ceux sur lesquels MCTS a obtenu les résultats que nous avons évoqué. Ces jeux sont en général des jeux

dits combinatoires, comme le sont les Échecs ou le Go. Il s'agit de jeux à deux joueurs à somme nulle, à information parfaite et dont le domaine est discret. D'autre part, même parmi ces jeux, ceux qui ont été étudiés se restreignent aux jeux dont le facteur de branchement est raisonnable, de l'ordre de la centaine.

La première différence entre ces jeux et la négociation tient au fait que cette dernière est un jeu à somme non nulle. C'est d'ailleurs pour trouver un accord mutuel qui soit profitable aux deux parties que les agents négocient les uns avec les autres. Cela influe sur la manière dont l'arbre est conçu. En effet, l'étape de sélection de l'arbre en UCT est faite selon le principe de minimax, qui consiste à supposer que l'adversaire cherche à minimiser notre utilité (pour maximiser la sienne).

Ensuite, la négociation est un jeu à information incomplète. Contrairement aux jeux à information imparfaite comme le poker, les jeux à information incomplète permettent de connaître tout l'historique du jeu, mais pas le type des joueurs. En l'occurrence en négociation le profil de préférence de l'adversaire est inconnu et fait même généralement l'objet d'une modélisation, comme nous l'avons exposé dans la Section 3.2.3.

Enfin, le domaine de la négociation est très particulier. Il dépend du domaine de négociation qui, s'il peut être discret – comme lorsque vous négociez la couleur de votre prochaine voiture avec votre famille – peut également être numérique lorsqu'on négocie une quantité voire continu dans certains cas. Cela a des conséquences sur le type de stratégies employé, comme nous l'avons exposé dans la Section 5.2.1.2. Même dans le cas où le domaine est discret, il est en général le produit cartésien des différents attributs qui le composent, et peut donc être extrêmement large. Pour 10 attributs de 10 valeurs, la taille du domaine et donc le facteur de branchement à chaque étape est de 10^{10}.

5.2.3 Discussion

Les méthodes de Monte-Carlo ont été appliquées à un grand nombre de domaines, et ont rencontré beaucoup de succès en IA pour les jeux. Comme nous l'avons déjà indiqué, la variante adaptée au domaine des jeux, MCTS a remporté nombre de victoires ces derniers temps. Étant donné que la négociation peut être décrite comme un jeu extensif avec un facteur de branchement important, il semble que les méthodes de Monte-Carlo pourraient obtenir de bons résultats dans ce domaine comme elles en ont eu pour les autres jeux. Cela est corroboré par les travaux de [de Jonge et Zhang, 2017], qui a conçu un agent de négociation pour les compétitions de *General Game Playing* et sur de petits domaines.

Nous nous intéressons dans le cadre de cette thèse à des domaines plus larges et complexes. Les particularités de la négociation rendent ainsi impossible l'application d'une implémentation de Monte-Carlo telles que celles qui ont servi pour [de Jonge et Zhang, 2017], pour le Go ou pour le Skat. Notre agent de négociation basé sur Monte Carlo devra donc en définitive utiliser des techniques qui seront davantage spécifiques à la négociation elle-même et laisser de côté UCT pour utiliser des méthodes comme le *progressive widening*. L'aspect jeu à information incomplète de la négociation force également à utiliser d'autres techniques, comme la modélisation d'adversaire pour inférer l'utilité de l'adversaire. Ce sont ces techniques que nous étudions dans la Section 5.3.

5.3 Un *Monte Carlo-based Negotiation Agent* (MoCaNA)

Comme nous l'avons expliqué dans la section précédente, la négociation est un jeu particulier et notre agent va donc devoir s'adapter en utilisant à la fois des techniques issues de la négociation automatique et de l'IA pour les jeux. Dans cette section, nous présentons notre stratégie de négociation automatique basée sur MCTS mais qui prend en compte ces particularités. L'architecture générale des différents modules de notre agent est présentée dans la Figure 5.3. Elle comporte les trois modules de BOA. La stratégie d'offre implémente MCTS mais utilise le module de modélisation d'adversaire, qui comporte deux sous-modules : l'un infère l'utilité de l'adversaire, l'autre modélise sa stratégie d'offre. Le dernier module, celui de stratégie d'acceptation, effectue une comparaison entre la proposition de l'adversaire et la proposition générée par la stratégie d'offre. Les différents sous-modules de l'agent ainsi que les interactions entre eux sont décrits dans la suite de cette section.

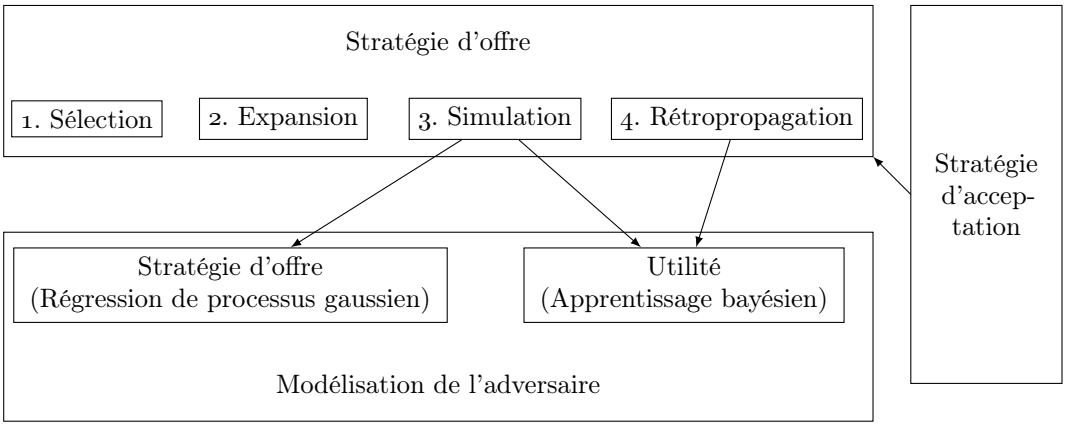


FIGURE 5.3 – Interaction des modules de MoCaNA

Nous proposons une approche originale, tant en ce qui concerne la modélisation d'adversaires, où nous modélisons à la fois l'utilité et la stratégie des agents qu'en ce qui concerne la stratégie d'offre, qui s'appuie sur MCTS. Ces différents modules n'ont pas de contrainte en ce qui concerne la nature des attributs, le fait qu'ils soient continus, entiers ou catégoriels. Notre implémentation de MCTS est capable de traiter indifféremment ces différents attributs, et nous proposons une extension des techniques de modélisation existants afin de les étendre au cas des attributs catégoriels (aussi appelés qualitatifs ou discrets). Les attributs catégoriels sont composés d'un nombre fini de valeurs possibles. Cela inclut par exemple le donneur d'ordre d'une facture. Dans les sections suivantes, nous introduisons les modules de modélisation d'adversaires, de stratégie d'offre et de stratégie d'acceptation de notre agent. Comme nous l'avons fait dans le Chapitre 3, nous nous appuyons sur le formalisme BOA, et proposons dans l'ordre nos modules de modélisation d'adversaires, notre stratégie d'offre et enfin notre stratégie d'acceptation.

Afin d'illustrer notre stratégie, nous proposons dans la suite de cette section l'exemple suivant : deux agents cherchent à négocier la vente d'un lot de factures. Dans notre exemple, deux agents, un fournisseur 1 et un investisseur 2 négocient un lot composé de plusieurs factures. La négociation porte sur le taux de rabais, mais aussi sur les attributs des factures contenues dans le lot. On suppose que les deux agents négocient à montant nominal fixé (la somme des montants des factures du lot est fixé). Le fournisseur fournit deux entreprises, et possède par conséquent des factures de deux donneurs d'ordre différents, pour deux types de produits. Les attributs de la négociation sont :

- le donneur d'ordre, un attribut catégoriel dont la valeur peut être soit a soit b,
- le type de produits vendus, un attribut catégoriel dont la valeur peut être soit p soit q,
- le taux de rabais (en pourcentage), une variable quantitative dont on restreint la valeur entre 1% et 11%.

L'historique de la négociation représenté dans la Figure 5.4. C'est un historique de ce type qui sera utilisé par les méthodes de modélisation de l'adversaire décrites dans cette section.

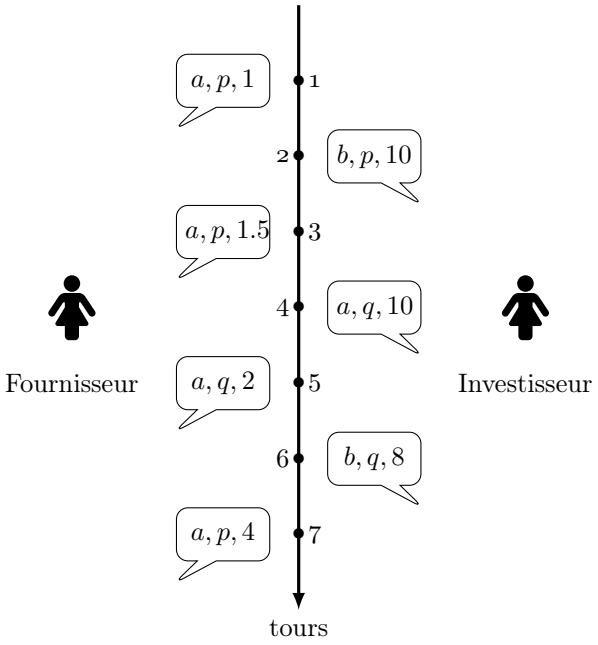


FIGURE 5.4 – Exemple de négociation multi-attribut

5.3.1 Modélisation d'adversaire

La modélisation d'adversaire est utilisée à plusieurs fins par notre agent. La modélisation de l'utilité de l'adversaire en particulier est utilisée pour pallier l'absence d'informations sur le type de l'adversaire – jeu à information incomplète. Elle est aussi utilisée pour améliorer les simulations. La modélisation de la stratégie d'offre de l'adversaire n'est utilisée que pour améliorer la pertinence des simulations.

Étant données les contraintes que nous nous sommes imposées – pas de suppositions sur la fonction d'utilité de l'adversaire, sur une borne ni sur le type d'attributs dont est composé le domaine de négociation – nous avons choisi des méthodes qui font le moins de suppositions possibles sur l'utilité ou la stratégie de l'adversaire. Nous proposons donc d'utiliser la régression de processus gaussien pour la modélisation de la stratégie, ce qui nous fournira en outre une gaussienne et permettra à notre MCTS de générer des simulations différentes, même en partant du même nœud, en faisant des tirages aléatoires dans les gaussiennes constituant la sortie de ce modèle. Quant au modèle d'utilité, nous utilisons celui fourni par [Hindriks et Tykhonov, 2008] parce qu'il fait peu de suppositions sur l'utilité de l'adversaire.

Nous étendons ces techniques au cas catégoriel afin de donner à notre agent la possibilité de négocier sur tous les domaines de négociation. Nous conservons cependant une contrainte généralement acceptée

dans le domaine de la négociation automatique : la supposition que les attributs sont indépendants les uns des autres, et que l'utilité d'un agent peut s'exprimer comme une somme pondérée des différents attributs. Cette supposition nous permet de traiter indépendamment les différents attributs, et de supposer que la fonction d'utilité peut s'exprimer comme la somme pondérée de la fonction d'utilité attribuée à chaque attribut.

5.3.1.1 Modélisation de la stratégie d'offre

Le but de la modélisation de stratégie d'offre est de prédire les futures propositions faites par l'adversaire. En particulier, le but est de prédire la prochaine proposition faite par l'agent au tour x_* de la négociation. Parmi les méthodes de modélisation de stratégie d'offre, celles qui reposent sur l'analyse de séries temporelles sont particulièrement adaptatives, puisqu'elles ne font aucune présupposition sur la stratégie de l'adversaire. En particulier, elles permettent de modéliser la stratégie adverse qu'il y ait ou non une borne à la négociation. La régression de processus gaussiens fait partie de ces méthodes. Elle permet de représenter la supposé prochaine proposition de l'adversaire à l'aide d'une gaussienne plutôt que d'une simple valeur. Cela est particulièrement utile dans le cadre d'une stratégie basée sur MCTS : chaque fois qu'une simulation est faite, une nouvelle valeur peut être générée par la gaussienne. Des valeurs différentes sont donc utilisées d'une simulation sur l'autre. De plus, ces valeurs sont générées selon la moyenne et l'écart type de la gaussienne, et correspondent donc aux valeurs que l'agent estime possibles, pondérées par leur probabilité étant données les propositions qu'il a reçues. L'ensemble des simulations modélisera bien l'adversaire, et prendra en compte les nuances liées à l'incertitude du modèle. Ce modèle, que nous avons brièvement présenté en Section 3.2.1.2, a été utilisé avec succès par [Williams *et al.*, 2011, Chen et Weiss, 2012, Williams *et al.*, 2013] et lui a permis de gagner les négociations contre d'autres agents, en particulier dans des domaines de négociation très larges.

Nous considérons les propositions de l'adversaire comme une série temporelle et essayons de prédire les suivantes. À cette fin, nous utilisons la régression de processus gaussien, telle que décrite par [Williams *et al.*, 2011] et l'étendons au cas des variables catégorielles en utilisant la classification basée sur les processus gaussiens décrit par [Williams et Barber, 1998]. Le but de cette technique est d'inférer deux éléments à partir des propositions connues : la valeur la plus probable pour la prochaine proposition et son écart type. Cette méthode ne fait aucune supposition sur la fonction de stratégie utilisée par l'adversaire.

Processus gaussien Les processus gaussiens peuvent être vus comme l'extension de la notion de loi normale multivariée aux fonctions, ou comme l'extension d'une fonction à un espace probabilisé.

Définition 5.1 (Loi normale multivariée)
Soit X une variable aléatoire à valeur dans $\mathbb{R}^n$. Alors, on dit que $\mathbf{X} = (X_1, \dots X_n)$ suit la loi normale multivariée de moyenne $\mu = (\mu_1, \dots, \mu_n)$ et de matrice de covariance Σ et on note $\mathbf{X} \sim \mathcal{N}(\mu, \Sigma)$ si et seulement si toute combinaison linéaire des X_i suit une loi normale :

$$\forall (a_1, \dots a_n) \in \mathbb{R}^n, \exists (\mu', \sigma) \in \mathbb{R}^2 : \sum a_i X_i \sim \mathcal{N}(\mu', \sigma) \tag{5.2}$$

Lorsqu'une variable aléatoire $\mathbf{X}$ suit une loi normale multivariée, chacune de ses composantes suit une loi gaussienne. Cela se vérifie en choisissant tous les a_i nuls à l'exception d'un seul dans la Définition 5.1. Cependant, dans le cas général une variable aléatoire gaussienne n'est pas la simple adjonction de gaussiennes indépendantes. Les variables aléatoires qui la composent ne sont pas nécessairement

indépendantes. Dans la Figure 5.5, nous représentons une loi normale bivariée dont les composantes sont corrélées positivement, les couleurs représentent la probabilité sur l'ensemble de l'espace. Lors d'un tirage sur cette loi, il y a davantage de probabilités d'obtenir des valeurs importantes sur les deux composantes qu'obtenir une valeur importante sur une des composantes, et une valeur faible sur l'autre. Cela se traduit dans la matrice de covariance Σ de la loi. Soit une variable $\mathbf{X} \sim \mathcal{N}(\mu, \Sigma)$ à valeurs dans $\mathbb{R}^n$, alors $\Sigma = (\Sigma_{ij})_{(i,j)\in[\![1,n]\!]^2}$ vérifie :

$$\Sigma_{ij} = \begin{cases} \sigma_{X_i} \text{ si } i = j \\ \rho(X_i, X_j) \text{ sinon} \end{cases} \tag{5.3}$$

où $\rho(X_i, X_j)$ est la corrélation entre les variables X_i et X_j.

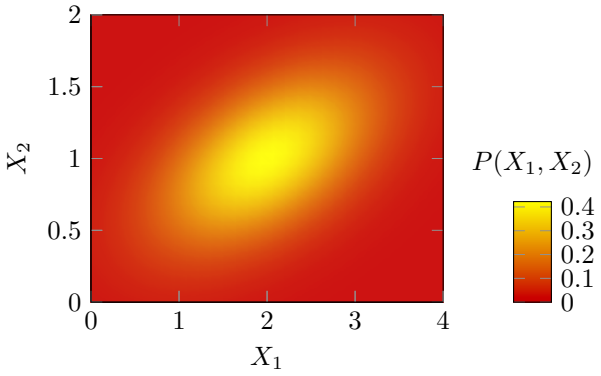


FIGURE 5.5 – Densité de probabilité d'une loi normale bivariée

Avant de décrire les processus gaussiens, nous énonçons le Théorème 5.1 portant sur les lois normales multivariées et qui nous servira par la suite. Ce théorème est notamment démontré par [Anderson, 2003]

Théorème 5.1 (Loi conditionnelle d'une loi normale multivariée)

Soit $\mathbf{X} = (X_1, \ldots, X_n)$ *une variable aléatoire multivariée telle que* $\mathbf{X} \sim \mathcal{N}(\mu, \Sigma)$ *avec* $\mu = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix}$ *et* $\Sigma = \begin{pmatrix} \Sigma_{11} & \ldots & \Sigma_{1n} \\ \vdots & & \vdots \\ \Sigma_{n1} & \ldots & \Sigma_{nn} \end{pmatrix}$*, symétrique définie positive. Soit* $A \subset [\![1, n]\!]$*, on note* $\mathbf{X}_A$ *la variable composée des composantes* X_i *où* $i \in A$*, et* $\mathbf{X}_{-A}$ *la variable composée des composantes* X_i *où* $i \notin A$*. Quitte à permuter les éléments de* $\mathbf{X}$*, on peut écrire :*

$$\mathbf{X} \sim \mathcal{N}\left(\begin{pmatrix} \mu_A \\ \mu_{-A} \end{pmatrix}, \begin{pmatrix} \Sigma_{AA} & \Sigma_{A-A} \\ \Sigma_{-AA} & \Sigma_{-A-A} \end{pmatrix}\right) \tag{5.4}$$

Alors, la loi conditionnelle de $\mathbf{X}_A$ *sachant que* $\mathbf{X}_{-A} = \mathbf{x}$ *vérifie :*

$$\mathbf{X}_{A|\mathbf{X}_{-A}=\mathbf{x}} \sim \mathcal{N}(\mu^A(\mathbf{x}), \Sigma_{A|-A}) \tag{5.5}$$

où :

$$\mu^A(\mathbf{x}) = \mu_A + \Sigma_{A-A}\Sigma^{-1}_{-A-A}(\mathbf{x} - \mu_{-A}) \tag{5.6}$$

$$\Sigma_{A|-A} = \Sigma_{AA} - \Sigma_{-A-A}\Sigma^{-1}_{-A-A}\Sigma_{-AA} \tag{5.7}$$

Un processus gaussien est une extension du cas multivarié au cas des fonctions. L'objectif de la régression de processus gaussien est d'obtenir des fonctions lisses (*smooth*). On considère chaque valeur de l'historique comme la réalisation d'une variable aléatoire gaussienne de moyenne nulle. On fait la supposition que plus un point est proche d'un autre, plus les valeurs de la fonction en ces points sont corrélées. On représente donc la covariance entre les valeurs de la fonction par une fonction de proximité entre les points. Cette fonction est la fonction de covariance, ou *kernel* k, et associe à deux valeurs du domaine de définition de la fonction une valeur de $[0, 1]$ où 0 représente deux points infiniment lointains et 1 deux points confondus. Ce *kernel* est en général une fonction paramétrée. Ces paramètres sont appris en utilisant une méthode d'optimisation afin de trouver les valeurs qui minimisent la distance entre les valeurs de l'historique et les valeurs prédites par le modèle.

Si on suppose trois points connus $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ sur la courbe d'une fonction f, on a donc selon ce modèle :

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} k(x_1,x_1) & k(x_1,x_2) & k(x_1,x_3) \\ k(x_2,x_1) & k(x_2,x_2) & k(x_2,x_3) \\ k(x_3,x_1) & k(x_3,x_2) & k(x_3,x_3) \end{pmatrix}\right) \tag{5.8}$$

Le but de la régression est de déterminer $y_* = f(x_*)$ pour un nouveau point x_*. On fait la même supposition sur f_* que nous l'avons fait pour f_i :

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mathbf{0} \\ 0 \end{pmatrix}, \begin{pmatrix} k(x_1,x_1) & k(x_1,x_2) & k(x_1,x_3) & k(x_1,x_*) \\ k(x_2,x_1) & k(x_2,x_2) & k(x_2,x_3) & k(x_2,x_*) \\ k(x_3,x_1) & k(x_3,x_2) & k(x_3,x_3) & k(x_3,x_*) \\ k(x_*,x_1) & k(x_*,x_2) & k(x_*,x_3) & k(x_*,x_*) \end{pmatrix}\right) \tag{5.9}$$

Notons qu'une fois cela fait, il est possible de déterminer cette valeur pour tout x du domaine de définition de f. On obtient donc un faisceau de fonctions. Lorsque f est à valeurs dans $\mathbb{R}$, il est possible de représenter ce faisceau de fonctions comme nous l'avons fait dans la Figure 5.6. Cette figure représente la densité de probabilité sur les valeurs de y résultant de la régression de processus gaussien. La régression a été effectuée avec les points pour $x \in \{1, 2, 4, 5, 6\}$.

Dans la suite de cette section, nous traitons séparément les cas des variables continues et catégorielles. Les variables catégorielles demandent un traitement supplémentaire, afin de traduire des valeurs non ordonnées en valeurs numériques. On sépare donc la matrice $\mathbf{y}$ en une première matrice contenant les valeurs numériques $\mathbf{y}_n$ et une seconde contenant les valeurs catégorielles $\mathbf{y}_c$. On traite ces deux matrices séparément.

Attribut numérique Plus formellement, la méthode procède ainsi. On commence par calculer la matrice de covariance, qui représente la proximité entre les tours de la séquence.

Soit

$$K = \begin{pmatrix} k(x_1,x_1) & \dots & k(x_1,x_n) \\ \vdots & & \vdots \\ k(x_n,x_1) & \dots & k(x_n,x_n) \end{pmatrix} \tag{5.10}$$

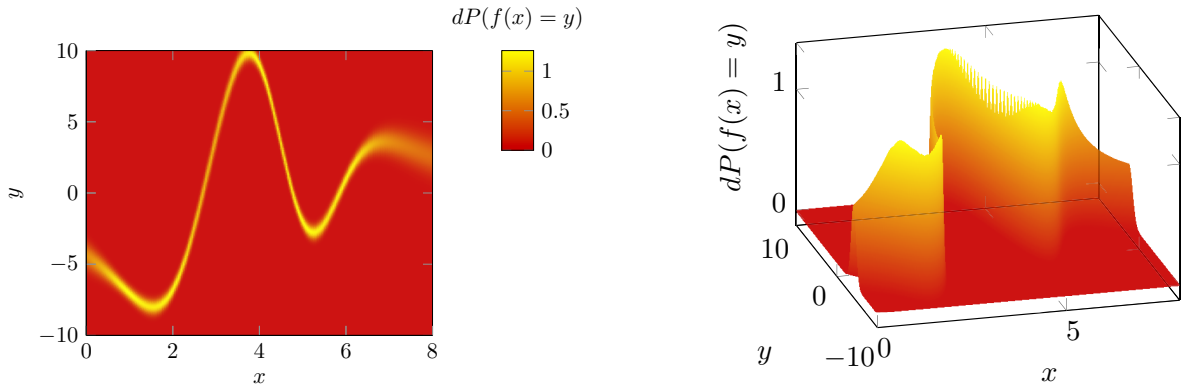


FIGURE 5.6 – Résultat d'une régression de processus gaussien

on calcule ensuite la distance entre le tour dont on veut estimer la proposition et les tours précédents :

$$K_* = (k(x_*, x_1), \ldots, k(x_*, x_n)) \tag{5.11}$$

en utilisant le Théorème 5.1, on peut calculer :

$$\overline{y_*} = K_* K^{-1} \mathbf{y} \tag{5.12}$$

$$\sigma_*^2 = \mathrm{Var}(y_*) = K_{**} - K_* K^{-1} K_*^\top \tag{5.13}$$

où $K_{**} = k(x_*, x_*)$. Nous identifions y_* à une variable aléatoire gaussienne de moyenne $\overline{y_*}$ et d'écart type σ_*.

Une des composantes les plus importantes de la régression de processus gaussien est la détermination du *kernel*. Parmi les *kernels* les plus communs, on retrouve les fonctions à base radiale, les fonctions *rational quadratic*, le *kernel* de Matérn et le *kernel exponential sine squared*. Nous décrivons ces fonctions dans les Équations (5.14) à (5.17). [Williams *et al.*, 2011] utilisent le *kernel* de Matérn et indiquent que c'est le mieux adapté à leur cas.

Dans notre cas, la distance entre propositions est déterminée par le seul élément connu par avance, à savoir le tour auquel il aura lieu. La Figure 5.7 représente les kernels précédemment cités. En abscisse se trouve la distance entre les tours et en ordonnée la distance calculée. Les kernels étant en général des fonctions paramétrées, les paramètres des courbes sont données dans la légende. Le kernel de Matérn (paramétré comme indiqué en légende) considère par exemple que la proposition précédente est trois fois et demi plus « proche » que celle qui est faite deux tours avant. En effet, $k_{Matérn}(1) = 0.48$ et $k_{Matérn}(2) = 0.14$.

Fonction à base radiale

$$k(x, x') = \exp\left(-\frac{1}{2} \cdot \left(\frac{x - x'}{l}\right)^2\right) \tag{5.14}$$

où l est un méta paramètre.

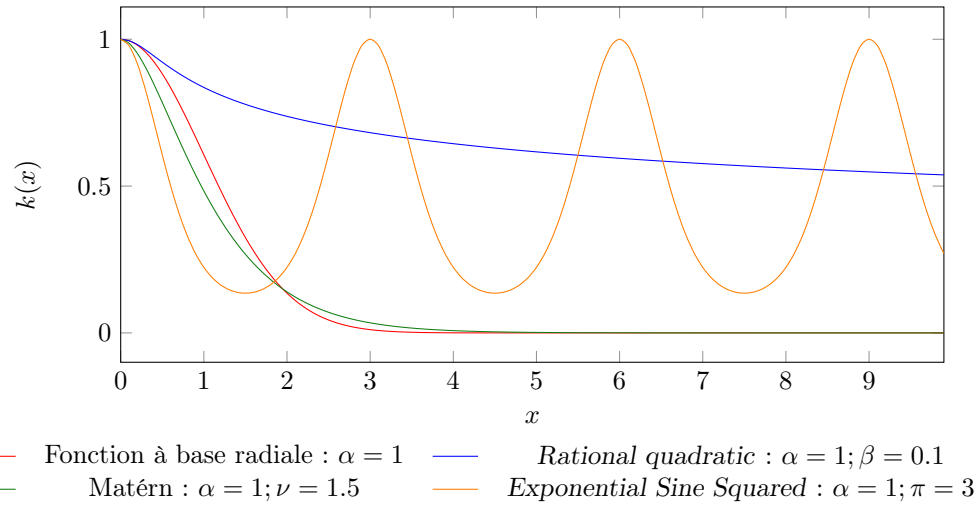


FIGURE 5.7 – Représentation des *kernels*

Rational quadratic

$$k(x, x') = \left(1 + \frac{(x - x')^2}{2 \cdot \alpha \cdot \beta^2}\right)^{-\alpha} \tag{5.15}$$

où α et β sont des méta paramètres.

Matérn

$$k(x, x') = \sigma^2 \cdot \frac{2^{1-\nu}}{\Gamma(\nu)} \cdot \frac{\sqrt{2\nu}\sqrt{(x' - x)^2}}{\alpha} \cdot K_\nu \left(\sqrt{2\nu}\frac{d}{\alpha}\right) \tag{5.16}$$

où α et ν sont des méta paramètres, σ est la variance des points et K_ν est la fonction modifiée de Bessel de second type, ainsi que présentée notamment par [Bowman, 2012].

Exponential sine squared

$$k(x, x') = \exp\left(-2 \sin\left(\frac{\pi}{\alpha} \cdot \frac{(x - x')^2}{\pi}\right)^2\right) \tag{5.17}$$

où π et α sont des méta paramètres.

Nous avons testé ces quatre *kernels* sur plusieurs négociations entre agents. Le Tableau 5.1 donne la table des résultats de la régression de processus gaussien pour chacun de ces *kernels*. Nous avons fait ces tests dans le contexte de l'ANAC 2014, où les préférences ne sont pas linéaires, et avec les finalistes de l'ANAC. Nous avons généré aléatoirement 25 sessions, et avons modélisé les deux agents, obtenant ainsi 50 modélisations au total. La table montre la distance euclidienne moyenne entre les propositions faites et le résultat de l'apprentissage. Plus la valeur est basse, plus la prédiction est proche de la véritable série.

Le *kernel* donnant le meilleur résultat est le *Rational quadratic*. C'est donc celui-ci que notre agent utilise.

Attribut catégoriel Le cas des attributs catégoriels est beaucoup plus complexe que celui des attributs numériques. Nous en donnons une introduction dans ce paragraphe, mais le lecteur intéressé

Kernel	Distance moyenne
Fonction à base radiale	43.288
Rational quadratic	**17.766**
Matérn	43.228
Exponential sine squared	22.292

TABLE 5.1 – *Kernels* pour la régression de processus gaussiens et distance moyenne aux vraies propositions

pourra se reporter à [Rasmussen et Williams, 2006b]. Le passage des attributs continus aux attributs catégoriels se fait comme le passage de la régression linéaire à la régression logistique. Nous traitons le cas binaire dans un premier temps, puis faisons l'extension au cas général à partir de ce dernier dans un second temps.

Dans le cadre de la classification binaire, on considère un attribut y prenant pour valeurs $\{0, 1\}$. L'objectif est, étant donné un historique composé de 0 et de 1 durant les tours $\mathbf{X}$, de déterminer la probabilité π_* que l'attribut prenne la valeur 1 au tour x_*. Pour déterminer π_*, on utilise une régression de processus gaussien, dont le résultat est passé par une fonction sigmoïde σ. Les fonctions les plus utilisées sont la fonction logit σ_{logit} et la fonction probit σ_{probit} :

$$\sigma_{probit}(x) = \int_{-\infty}^{x} \frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}} dt \tag{5.18}$$

$$\sigma_{logit}(x) = \frac{1}{1 + e^{-x}} \tag{5.19}$$

Ces fonctions sont aussi utilisées en régression logistique. Elles permettent d'associer à une fonction à valeurs dans $\mathbb{R}$ une fonction à valeurs dans $[0, 1]$ séparant « bien » les différentes classes. La Figure 5.8 montre la fonction logit. En utilisant une fonction à valeurs dans $\mathbb{R}$ en entrée, elle permet d'associer aux valeurs négative à forte valeur absolue une probabilité quasi nulle et aux valeurs très hautes une probabilité proche de 1. Les valeurs autour de 0 sont associées à une probabilité intermédiaire.

Il est possible de déterminer la probabilité que y prenne la valeur 1 grâce à la formule en calculant la moyenne des sigmoïdes appliquées aux fonctions du réseau obtenues par régression de processus gaussien, pondérées par la probabilité associée à chacune de ces fonctions. Puisque le faisceau de fonctions correspond à une densité, la probabilité est le résultat de l'intégration sur ces fonctions :

$$\pi_* = P(y = 1) = \int \sigma(f) p(f|\mathbf{X}, \mathbf{y}, x_*) df \tag{5.20}$$

où $\mathbf{X}, \mathbf{y}$ forme l'historique.

Le problème est que cette intégrale n'est pas calculable. En particulier, $p(f|\mathbf{X}, \mathbf{y}, x_*)$ se calcule au moyen de la loi de Bayes. L'intégrale qui en résulte fait elle-même appel à une fonction sigmoïde,

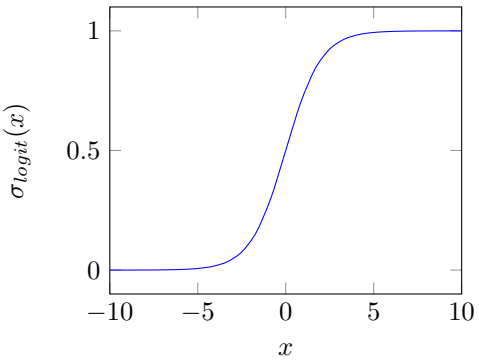


FIGURE 5.8 – Fonction logit

qui rend l'intégrale incalculable. Il est possible d'approximer le calcul par une méthode numérique, par exemple la méthode de Laplace décrite par [Williams et Barber, 1998]. Cette méthode consiste à approximer la fonction par une gaussienne, ce qui la rend calculable. L'utilisation de cette fonction demande cependant de déterminer la fonction maximisant la vraisemblance des valeurs obtenues, c'est à dire les valeurs $\mathbf{z} = \left(\hat{f}(X_1), \ldots, \hat{f}(X_n)\right)$ minimisant la distance entre $\sigma(\mathbf{z})$ et $\mathbf{y}$. Ces valeurs peuvent se calculer en utilisant la méthode de Newton. La complexité globale de cette méthode est en $\mathcal{O}(n^3)$, où n est le nombre de valeurs connues.

Intéressons-nous maintenant au cas multi-attributs. On suppose qu'il y a n attributs catégoriels, et on note $(C_k)_{k\leq n}$ le nombre de valeurs (ou de catégories) de l'attribut k. On cherche à établir la probabilité de chaque valeur possible pour chaque attribut. On fait la supposition que la valeur obtenue peut être exprimée comme une probabilité dépendant de diverses fonctions, chacune étant utilisée pour une valeur. Pour chaque variable k, il est possible de construire un vecteur de fonctions dont chaque valeur correspond à une valeur donnée. On note ce vecteur

$$\mathbf{f}_i = \begin{pmatrix} f_i^1 \\ \vdots \\ f_i^C \end{pmatrix} \tag{5.21}$$

La probabilité globale d'une valeur c connaissant la fonction de probabilité $\mathbf{f}_i$ pour chaque valeur est déterminée grâce au critère du *softmax* exprimé comme :

$$p(y_i = c|\mathbf{f}_i) = \frac{\exp(f_i^c)}{\sum\limits_{c' \leq C_i} \exp(f_i^{c'})} \tag{5.22}$$

La détermination des probabilités se fait comme dans le cas binaire. Une fois la probabilité obtenues pour chaque valeur, il reste à déduire les $p(y_* = c|\hat{\mathbf{f}}, x_*)$. Cette valeur ne peut être obtenue simplement en utilisant l'Équation (5.22), car la covariance entre les valeurs, représentée par Σ peut changer cet ordre (voir [Rasmussen et Williams, 2006b] pour un exemple). On va donc utiliser une méthode de Monte Carlo : faire un certain nombre de tirages pour les $\left(\hat{\mathbf{f}}_i\right)_{i\in[\![1,C]\!]}$ et déterminer la loi de probabilité des différentes valeurs en fonction du résultat des tirages.

Malgré sa complexité, cette technique s'est montrée particulièrement efficace dans les problèmes de classification, comme le montrent [Nickisch et Rasmussen, 2008].

Notons que puisque nous supposons que les attributs sont indépendants les uns des autres, il est possible de modéliser les attributs séparément, et de prédire lequel sera le suivant indépendamment de ce qui est prédit par les autres ; notre agent utilise cette approche.

5.3.1.2 Modèle du profil de préférence

Les méthodes de modélisation de l'utilité adverse sont moins nombreuses, et on notamment été décrites dans la Section 3.2.3. Les méthodes à base de fréquence ne permettent pas de modéliser des préférences sur un domaine continu, et les autres méthodes, l'apprentissage bayésien et la régression non linéaire, font toutes deux la supposition que l'adversaire fait des concessions à un certain rythme. De plus, la régression non linéaire est utilisée dans le cas où l'agent a une connaissance *a priori* de la famille de fonctions à laquelle appartient la stratégie. L'apprentissage bayésien décrit par [Hindriks et Tykhonov, 2008] fait une supposition moins forte, en supposant que l'agent fait une concession à un rythme constant. Nous utilisons donc cette méthode.

Cette méthode repose sur l'approximation de l'utilité de l'adversaire par une somme pondérée de fonctions triangulaires. Ces fonctions sont définies dans la Définition 5.2 et illustrées dans la Figure 5.9

Définition 5.2 (Fonction triangulaire)
Soit un attribut quantitatif A, ayant pour minimum et maximum respectivement min_A et max_A. Alors, une fonction triangulaire t de A est une fonction de $[0,1]^{[min_A, max_A]}$ vérifiant :

- *il existe a dans $[min_A, max_A]$ tel que $t_a(min_A) = t_A(max_A) = 0$, $t_A(a) = 1$ et t_A est linéaire entre min_A et a et entre a et max_A ou*
- *t_a est linéaire et $t_a(min_A) = 0$ et $t_a(max_A) = 1$ ou*
- *t_a est linéaire et $t_a(min_A) = 1$ et $t_a(max_A) = 0$.*

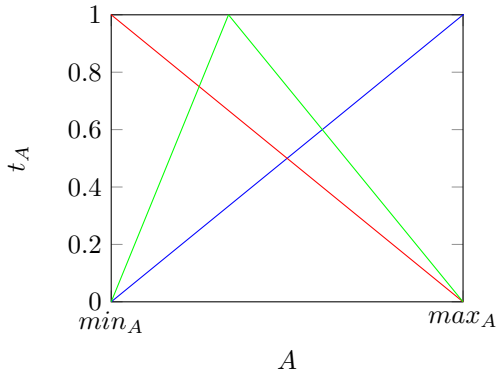


FIGURE 5.9 – Exemples de fonctions triangulaires

La méthode se divise en deux étapes :

1. D'abord, un certain nombre d'hypothèses sont générées. Ces hypothèses sont composées d'une somme pondérée de fonctions triangulaires. Chaque attribut se voit attribuer un rang et une fonction triangulaire.
2. Ensuite, en fonction des propositions reçues, les probabilités de chaque hypothèse sont mises à jour avec la loi de Bayes.

Algorithme 4 Apprentissage bayésien de l'utilité - cas numérique

function GÉNÉRERHYPOTHÈSE($n \in \mathbb{N}$ nombre d'attributs, $min \in \mathbb{N}^n$ liste des minima des attributs, $max \in \mathbb{N}^n$ liste des maxima des attributs)
  générer $\tau : [\![1, n]\!] \to [\![1, n]\!]$ ▷ permutation pour assigner les rangs
  for $i \in [\![1, n]\!]$ **do**
    $w_i \leftarrow 2 \cdot \dfrac{\tau(i)}{n \cdot (n+1)}$
    $t \leftarrow alea(min, max)$ ▷ pointe du triangle
  end for
  return $p \in \mathbb{R}^n \mapsto \sum_{k \in [\![1,n]\!]} w_i \cdot \left(\mathbb{1}_{p_i \leq t_i} \cdot \dfrac{p_i - min_i}{t_i - min_i} + \mathbb{1}_{p_i > t_i} \cdot \dfrac{max_i - p_i}{max_i - t_i} \right)$
    ▷ où $p_i \mapsto \left(\mathbb{1}_{p_i \leq t_i} \cdot \dfrac{p_i - min_i}{t_i - min_i} + \mathbb{1}_{p_i > t_i} \cdot \dfrac{max_i - p_i}{max_i - t_i} \right)$ est la fonction triangulaire
end function
function ESTIMERUTILITÉ($n \in \mathbb{N}$:nombre d'hypothèses,$(b_j)_{j \leq m}$:séquence des offres reçues, $(\alpha, \sigma) \in \mathbb{R}$:paramètres du modèle)
  for $i \in [\![1, n]\!]$ **do**
    $t_i \leftarrow hypotheseAleatoire()$ ▷ $t[i]$ est une fonction hypothèse.
  end for
  for $j \in [\![1, m]\!]$ **do**
    for $i \in [\![1, n]\!]$ **do**
      $p_{i,j} = \dfrac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(t_i \cdot f(b_j) - u^*(b_j))^2}{2\sigma^2}}$ ▷ $p_{ij} = \mathbb{P}(b_j | u = t_i)$,$u^*(b_t) = 1 - \alpha t$
      $p'_{i,j} = \dfrac{p_i \cdot p_{i,j}}{\sum_{k \leq n} p_k \cdot p_{k,j}}$ ▷ $p'_{ij} = \mathbb{P}(u = t_i | b_j)$
      $p_i = p_i \cdot p'_{i,j}$ ▷ $p_i = \mathbb{P}\left(t_i | (b_j)_{j \in [\![1,m]\!]}\right)$
    end for
  end for
  return $\sum_{i \in [\![1,n]\!]} p_i \cdot t_i$
end function

L'utilité estimée d'un adversaire est la somme de ces hypothèses pondérée par la probabilité de chacune. Cette méthode a pour avantage de ne pas faire de supposition sur la stratégie de l'adversaire, mais elle suppose qu'il fait des concessions à un rythme globalement linéaire.

Attributs numériques Nous présentons formellement la méthode dans l'Algorithme 4.

Cet algorithme est justifiable par la loi de Bayes. En effet, en supposant que l'envoi d'une proposition est une gaussienne autour de l'utilité supposée de l'agent (selon un rythme fixe de α), la probabilité que l'adversaire envoie l'offre b_j sachant que l'hypothèse t_i est la bonne est :

$$\mathbb{P}(b_j | u = t_i) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(t_i(b_j) - u^*(b_j))^2}{2\sigma^2}} \tag{5.23}$$

où :

- $u^*(b_t) = 1 - \alpha t$ est l'utilité supposée de l'agent, où le rythme de concession est α,

- σ est un paramètre du modèle, et contrôle la confiance de l'agent dans son modèle de concession. Plus σ est petit, plus l'agent est confiant dans sa fonction u^*

Il est alors possible d'utiliser la loi de Bayes :

$$\mathbb{P}(t_i|b_j) = \frac{\mathbb{P}(t_i)\mathbb{P}(b_j|t_i)}{\sum\limits_{k\leq n} \mathbb{P}(t_k)\mathbb{P}(b_j|t_k)} \tag{5.24}$$

Reprenons notre exemple, et déterminons une fonction d'utilité sur le taux de rabais. Supposons que cette fonction soit une simple fonction linéaire t dont le maximum se trouve en 1. On a alors $t(4) = \frac{4-1}{11-1} = 0.7$. On suppose également que $\sigma = 1$. Alors, la probabilité que l'agent fasse une proposition b avec un taux de 4% à son quatrième tour en supposant que sa fonction d'utilité est t est :

$$\mathbb{P}(b|t) = \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{1}{2}(0.7-0.8)^2} = 0.1596 \tag{5.25}$$

La probabilité de $u = t_i$, qui est identifiée à la probabilité conditionnelle étant donnés toutes les offres reçues. Elle est obtenue en faisant la supposition que toutes les offres sont indépendantes, et donc que la probabilité globale est le produit des probabilités conditionnelles en regard de chaque offre.

$$\mathbb{P}(t_i) = \mathbb{P}(t_i|b_1 \dots b_m) = \prod_{j\leq m} \mathbb{P}(t_i|b_j) \tag{5.26}$$

Attributs catégoriels Nous proposons d'étendre l'algorithme de [Hindriks et Tykhonov, 2008] en appliquant l'apprentissage bayésien aux attributs catégoriels. L'algorithme est très similaire à celui du cas numérique, à cela près que les hypothèses ne seront pas des fonctions triangulaires mais des utilités sur chaque valeur possible de l'attribut. L'unique changement est donc la fonction de génération des hypothèses que nous décrivons en Algorithme 5.

Algorithme 5 Apprentissage bayésien de l'utilité - cas catégoriel

function GÉNÉRERHYPOTHÈSE($n \in \mathbb{N}$ nombre d'attributs, $((V_i)_{i\leq n}$: liste des valeurs possibles pour chaque attribut)
  générer $\tau : [\![1, n]\!] \to [\![1, n]\!]$ ▷ permutation pour assigner les rangs
  $borne \leftarrow 1$
  μ ▷ Liste de liste vides, à remplir avec l'utilité des valeurs de chaque attribut
  for $i \in [\![1, n]\!]$ **do**
    $w_i \leftarrow 2 \cdot \frac{\tau(i)}{n \cdot (n+1)}$
    for $j \in [\![1, |V_i|]\!]$ **do**
      $\mu_i(j) \leftarrow$ alea$(0, borne)$ ▷ Utilité associée à la valeur j de l'attribut i
      $borne \leftarrow borne - p_i(j)$ ▷ Afin que les utilités des valeurs somment à 1
    end for
  end for
  return $p \in \mathbb{V}_i \mapsto \sum\limits_{k\in[\![1,n]\!]} w_i \cdot \mu_i(p_i)$
end function

Notons qu'il est possible de mêler des attributs catégoriels et numériques à condition de combiner les deux fonctions de génération d'hypothèses.

5.3.2 Une stratégie d'offre basée sur MCTS

Les méthodes de Monte-Carlo sont très adaptatives, et ont obtenu de bons résultats dans différents jeux, y compris des jeux à grand facteur de branchement (tel que le Go : [Silver *et al.*, 2016]). Elles sélectionnent un coup en donnant à chacun un score basé sur le résultat de simulations. Ces simulations utilisent un grand nombre de coups afin de couvrir autant de cas que possible. Notre but est de mesurer leurs performances dans le contexte de la négociation automatique et de concevoir un agent qui soit capable de négocier dans un contexte où le domaine de négociation est continu et où la présence d'une borne n'est pas pertinent.

5.3.2.1 MCTS sans élagage

Dans la suite de cette partie, nous présentons un module de stratégie d'offre qui exploite les techniques de modélisation d'adversaire présentées dans la Section 5.3.1. Cette stratégie étant basée sur une méthode de Monte-Carlo, nous commençons par décrire le fonctionnement de ces techniques.

Comme nous l'avons expliqué, MCTS est un algorithme général, qui demande à ce que chacune de ses stratégies soient implémentées. Dans le contexte de la négociation :

Sélection Pour cette étape, nous utilisons le *progressive widening*. La principale problématique posée par les domaines continus est le dilemme exploration/exploitation. Le critère du progressive widening peut s'exprimer ainsi : un nouveau nœud est étendu si et seulement si :

$$n_p^\alpha \geq n_c \tag{5.27}$$

où n_p est le nombre de fois où le parent a été simulé, n_c est le nombre d'enfants qu'il a, et α est un paramètre du modèle. S'il n'y a pas d'expansion, un nœud fils est sélectionné. Soit n le nombre total de simulations faites jusqu'alors (*i.e.* le nombre de fois où l'arbre a été visité), s_i le score du nœud i, nous sélectionnons alors le nœud i qui maximise :

$$W_i = \frac{s_i}{n_i + 1} + C \times n^\alpha \sqrt{\frac{\ln(n)}{n_i + 1}} \tag{5.28}$$

où C est également un paramètre du modèle. Comme nous l'avons montré dans la Définition 4.2, la négociation peut être décrite comme un jeu extensif. Or, les méthodes de Monte Carlo ont montré leur efficacité dans la résolution de tels jeux, comme l'ont notamment montré les travaux de [Chaslot *et al.*, 2008a, Kupferschmid et Helmert, 2007]. On identifie alors un coup du jeu, et donc un nœud de l'arbre de MCTS, à un message envoyé lors de la négociation. Notre domaine de négociation est potentiellement continu, et s'il ne l'est pas, il peut contenir un nombre très important de possibles propositions. C'est par exemple le cas des domaines de négociation de l'ANAC 2014 dont la taille du domaine de négociation varie entre 10^{10} et 10^{50} mouvements possibles, ce qui est trop important pour que chaque possibilité ne soit simulée ne serait-ce qu'une fois.

Expansion La décision d'étendre un nouveau nœud est donc prise selon le critère du *progressive widening*. La valeur du nœud étendu est quant à elle choisie de manière aléatoire.

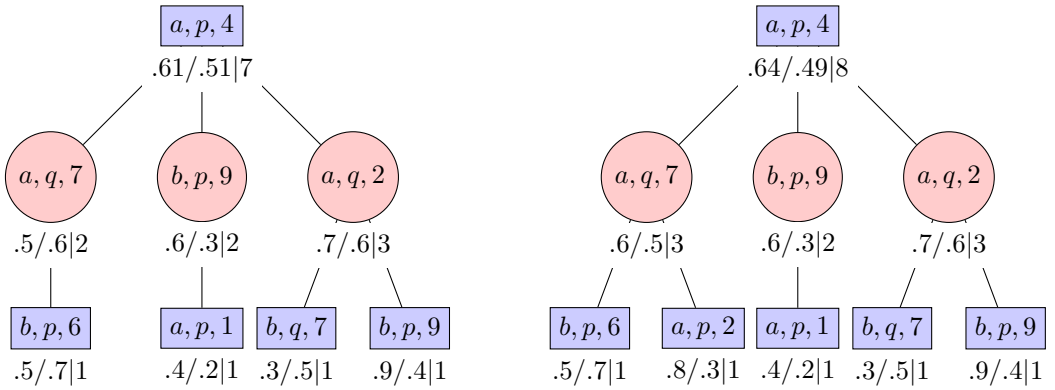
Simulation Nous utilisons le modèle de stratégie de l'adversaire, comme le recommandent [Ponsen *et al.*, 2010], afin que chaque simulation soit aussi vraisemblable que possible. Nous utilisons la régression de processus gaussien pour modéliser la stratégie de l'adversaire et l'apprentissage bayésien décrits dans la Section 5.3.1.2 pour modéliser son utilité. En ce qui concerne

la stratégie d'acceptation de l'adversaire, nous utilisons la même stratégie d'acceptation que la nôtre. Elle est décrite dans la Section 5.3.3 : l'agent simulé accepte une proposition quand la proposition générée par le modèle de régression de processus gaussien lui apporte une moins bonne utilité que la dernière proposition qu'il a reçue.

Rétropropagation L'étape de rétropropagation utilise également le modèle d'utilité de l'adversaire. La négociation est un jeu à somme non nulle. Un agent rationnel ne va donc pas choisir la proposition qui minimise l'utilité adverse, mais celle qui maximise sa propre utilité. Le critère de sélection dans l'arbre MCTS prend notamment en compte le score. Lorsqu'on considère un nœud correspondant à un coup de l'adversaire, le nœud suivant est sélectionné selon son utilité. Il est donc nécessaire de connaître l'utilité de l'adversaire. Puisque cette information n'est pas accessible, nous utilisons un modèle de son utilité.

Enfin, MCTS fait des simulations et calcule l'utilité attendue de toute la négociation. Il a tendance à sous-estimer la valeur de la proposition qu'il ressort, et la probabilité que l'adversaire l'accepte (le principe d'un « tiens » vaut mieux que deux « tu l'auras »). Afin de prendre cet aspect en compte, nous calculons la moyenne avec l'utilité du coup s'il est accepté par l'adversaire. Nous renvoyons donc l'offre b^* parmi les fils de la racine maximisant :

$$b^* = \underset{b \in root.children}{\operatorname{argmax}} \left(\frac{utility(b) + score(b)}{2} \right) \tag{5.29}$$



(a) État initial (b) Après exécution d'une itération de MCTS

FIGURE 5.10 – Algorithme Monte-Carlo de MoCaNA

La Figure 5.10 donne une illustration d'un arbre MCTS construit pour notre exemple. Cet arbre correspond à une situation où 7 simulations ont déjà été réalisée. La Figure 5.10a montre l'état de l'arbre avant la réalisation de l'étape de MCTS. Pour notre exemple, on suppose que $\alpha = 0.5$ et $C = 0.5$. La racine de l'arbre a été sélectionnée 7 fois. On a donc $n_p^\alpha = 2.65 < 3 = n_c$. On sélectionne donc un nœud existant. On calcule les valeurs de W_i pour les trois fils de la racine. On numérote les nœuds de la gauche vers la droite. Pour $i = 1$, on a $n_1 = 2, n = 7, s_1 = 0.6$ et donc $W_1 = 1.27$. En procédant de même pour les autres nœuds, on obtient $W_2 = 1.23$ et $W_3 = 1.072$. On sélectionne donc le premier fils. On procède ensuite à nouveau au test déterminant s'il faut étendre un nouveau nœud. On obtient $n_p^\alpha = 1.414 > 1 = n_c$. On génère donc un nouveau nœud, de manière aléatoire, ici $a, p, 2$. On procède à une simulation en utilisant la modélisation de l'adversaire, et grâce à la modélisation de son utilité, la simulation renvoie un score de 0.8 pour MoCaNA et de 0.3 pour son adversaire. Ce score est rétropropagé en le moyennant avec le nœud $a, q, 7$ et la racine. Le résultat de ces étapes est représenté en Figure 5.10b.

5.3.2.2 Élagage

La fait que notre agent ne prenne pas en compte la valeur intrinsèque des nœuds étendus peut être traité différemment, et plus particulièrement directement dans MCTS. Comme nous l'avons exposé dans la Section 5.2.1.2, il est possible d'utiliser des connaissances sur le jeu pour élaguer certaines parties de l'arbre et ne pas les développer. Nous développons ci-après une stratégie sur un MCTS élagué, où les parties les moins prometteuses de l'arbre ne sont pas développées.

Par une valeur fixe Nous considérons deux cas. Dans le premier, nous supposons que notre agent souhaite atteindre une utilité minimale. Ce type de supposition a également été faite par d'autres agents, comme nous l'avons exposé dans le Chapitre 3. Toute branche où notre agent génère une proposition amenant une utilité inférieure à cette utilité seuil est supprimée. En terme d'implémentation, on ne génère que des offres dont l'utilité est supérieure à cette valeur seuil.

Selon les propositions de l'adversaire Lorsqu'il est impossible de déterminer une valeur seuil, typiquement parce que le domaine est peu connu, ou par crainte qu'une telle valeur ne soit pas acceptable pour l'adversaire, il est possible de se baser sur les propositions de l'adversaire pour déterminer un seuil qui évoluera au cours de la négociation. On se servira de la plus haute utilité offerte par l'adversaire pour déterminer un seuil. Cette valeur augmentant au cours du temps, le seuil augmentera lui aussi. Cette technique sera donc d'autant plus précise que de nombreuses propositions ont été reçues par l'adversaire.

D'autres agents utilisent les meilleures propositions faites par l'adversaire pour leur propre stratégie. C'est par exemple le cas de BraveCat, que nous avons évoqué dans le Chapitre 3 qui à la fin de la négociation envoie les propositions de son adversaire qui lui amène la plus haute utilité.

5.3.3 Stratégie d'acceptation

Nous utilisons la même stratégie d'acceptation que celle de DoNA, et qui est décrite par [Baarslag *et al.*, 2015b] et que nous avons évoquée à la Section 3.4.1. Cette stratégie semble appropriée à notre contexte : lorsque la dernière proposition est meilleure que celle générée par la stratégie d'offre, notre agent l'accepte.

5.3.4 Discussion

Notre agent utilise plusieurs techniques actuellement largement utilisées en intelligence artificielle : les méthodes de recherche arborescente de Monte Carlo, ainsi que l'apprentissage artificiel, utilisé pour modéliser l'adversaire. es méthodes permettent à notre agent de négocier dans un vaste ensemble de situations,quelle que soit la nature des attributs du domaine de négociation. De plus ces méthodes reposent sur peu de contraintes principalement du fait que les différents attributs du domaine sont indépendants.

Le fait d'utiliser une modélisation de la stratégie adverse apporte une plus-value à notre agent. Elle alourdit cependant le temps de calcul. En particulier, la modélisation de la stratégie de l'adversaire repose sur du calcul matriciel. Afin d'éviter un alourdissement trop important, nous effectuons l'ap-

prentissage une unique fois, à la réception de l'offre de l'adversaire. Une fois le modèle appris, il est utilisé pour tous les mouvements ultérieurs.

Comme nous l'avons indiqué, ces méthodes permettent à notre agent de négocier sous diverses conditions. Cela signifie notamment que notre agent ne tire pas parti des particularités du domaine. Face à des agents plus spécialisés, il est donc désavantagé.

5.4 Analyse expérimentale

Pour évaluer notre agent, nous le confrontons aux finalistes de l'ANAC 2014. Certains de ces agents sont décrits dans le Chapitre 3. Les compétitions de l'ANAC sont généralement menées en utilisant le *framework* GENIUS, et les agents ayant participé au tournoi sont fournis avec le *framework*.

Il est à noter que ces agents ont été spécifiquement conçus en vue d'obtenir de bons résultats dans le cadre de cette compétition. Ils exploitent donc autant que possible ses spécificités : une borne sous la forme d'une limite en temps ou en nombre de tours et un domaine de négociation discret quoique vaste. L'ANAC n'étant pas la finalité de cette thèse, nous avons conçus notre agent afin qu'il réponde à un contexte plus général, en adéquation avec notre cadre applicatif. En particulier, aucune hypothèse n'est faite sur une éventuelle borne ni sur le caractère discret ou continu du domaine. Cet état de fait est pénalisant pour MoCaNA dans le cadre de cette compétition. Néanmoins, dans la mesure où elle est une référence dans le domaine de la négociation automatique, nous présentons dans cette section les résultats des comparaison de notre agent avec ceux de l'ANAC.

Afin de compléter cette analyse expérimentale en nous comparant avec un agent dans les mêmes conditions que nous-même, nous implémentons de plus un adversaire basé sur une version du *Random Walker*, définie par [Baarslag et Hindriks, 2013] [1]. Cet agent n'a pas de module d'apprentissage, et fait des propositions aléatoires. Il n'accepte jamais une proposition de l'adversaire, ce qui pousse ce dernier à trouver le bon moment pour accepter.

5.4.1 Détails d'implémentation

En termes d'architecture, notre agent a été conçu pour être utilisé aussi bien dans GENIUS qu'en dehors. Nous avons implémenté des interfaces représentant le domaine d'utilité de l'agent et le protocole, ainsi qu'une classe qui implémente un historique. Cela forme, avec les différents modules de l'agent, un cadre suffisant pour que l'agent soit autonome et ne dépende pas du *framework*.

L'architecture de MoCaNA est implémentée en Java. Notre implémentation est modulaire et fait une claire distinction entre les modules sus-mentionnés. Nous utilisons la librairie JAMA (pour JAva MAtrix) pour la manipulation de matrices. Cette librairie est issue des *Java Numerics* du NIST, qui sont décrits dans [Jav, 2013]. En ce qui concerne les autres outils mathématiques, notamment l'optimisation, nous utilisons la librairie Apache Commons Math, décrite dans [Math, 2016]. Les services fournis par cette librairie, et en particulier la descente de gradient sont utilisés pour optimiser les paramètres de la régression de processus gaussien α et k. Chaque élément de l'historique est utilisé pour apprendre les deux modèles de l'adversaire. Pour la modélisation de l'utilité, nous générons 200 hypothèses aléatoires.

1. Bien que le nom puisse laisser à penser le contraire, cet agent n'implémente en aucun cas une *random walk*. Il s'agit d'un agent faisant des propositions aléatoires.

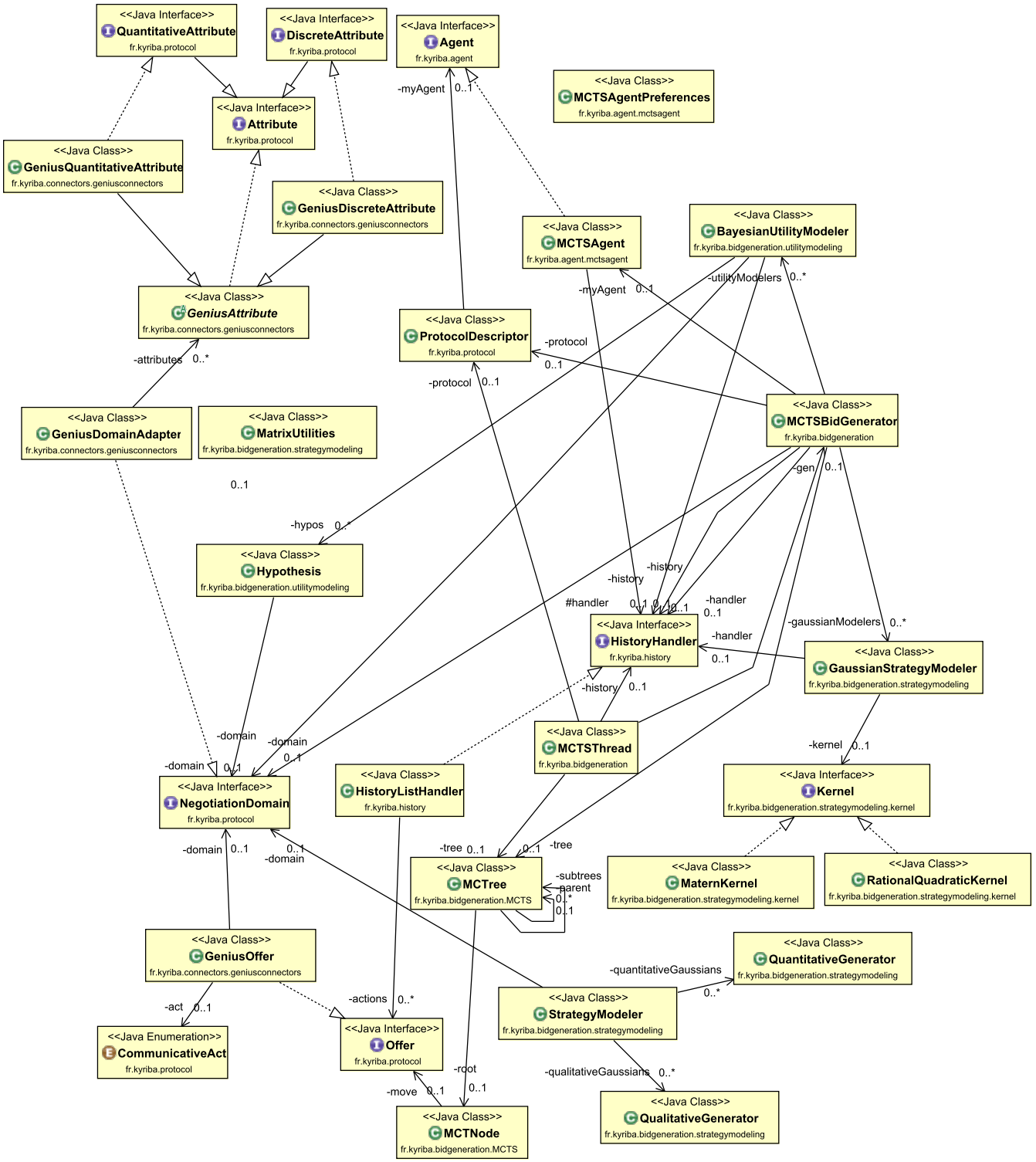


FIGURE 5.11 – Diagramme de classes de MoCaNA

Notre modèle compte cependant d'autres paramètres que ceux utilisés par la régression de processus gaussien. En ce qui concerne la modélisation de l'utilité, l'information de l'adversaire n'est pas connue. Il est donc impossible de valider en utilisant un algorithme d'optimisation. Nous utilisons les valeurs données par [Hindriks et Tykhonov, 2008], à savoir $\alpha = 0.05$. Nous fixons la valeur de σ empiriquement à $\sigma = 0.5$. En ce qui concerne l'élagage par une valeur fixe, nous utilisons une valeur cible de 0.8, qui permet à la fois d'être assez faible pour espérer un `accept` et assez forte pour obtenir une utilité supérieure à celle de l'adversaire.

Nous utilisons également les possibilités offertes par le multithreading. Nos expérimentations sont menées sur un processeur octothread de type i7 cadencé à 3.6MHz, ce qui nous permet de paralléliser le MCTS. L'arbre est synchronisé, et chaque thread fait tourner ses quatre étapes de l'algorithme. Puisqu'il est distribué sur plusieurs threads, l'exécution d'un algorithme MCTS n'attend pas qu'un autre soit fini et ait donc mis les scores l'arbre à jour pour démarrer. Cela permet de faire plus de simulations, mais chaque sélection se fait sur des scores qui ne sont pas forcément encore mis à jour. Cependant, comme le montrent [Chaslot *et al.*, 2008b], cette version parallélisée de MCTS est efficace malgré tout.

Cela est représenté sur le diagramme de classes de notre agent, représenté sur la Figure 5.11. On y retrouve la modélisation de l'apprentissage bayésien, sous la forme de deux classes, dont l'une représente à la fois les hypothèses numériques et les hypothèses catégorielles. La stratégie d'offres, `MCTSBidGenerator` utilise une représentation de l'arbre. Les tests sur les *kernels* ayant été réalisés en python, nous avons implémentés le *kernel* de Matérn, conseillé par [Williams *et al.*, 2011] et le *kernel Rational Quadratic*, qui donnait les meilleurs résultats dans notre cas. La modélisation de la stratégie adverse résulte en des instances de deux classes, une pour les attributs numériques, et une pour les attributs catégoriels. Ces deux classes sont définies par une méthode qui permet de générer une valeur sur la gaussienne résultant de la régression. L'adaptation à GENIUS est donnée sur la gauche du diagramme, et implémente des interface qui peuvent tout à fait être implémentées par un autre environnement, permettant d'adapter facilement notre agent à de nouveaux frameworks.

Au-delà de ces expérimentations, l'objectif de cet agent est qu'il soit intégré dans notre place de marché. Or, cette dernière est vouée à intégrer les protocoles présentés dans le Chapitre 4, mais aussi des protocoles sans limite, ni sur le temps ni sur le nombre de tours de négociation. De même, et comme nous l'avons évoqué, la négociation peut porter sur différents attributs des factures, dont certains sont continus, d'autres entiers et d'autres catégoriels. Les suppositions faites par l'ANAC ne collent donc pas à l'objectif fixé à notre agent. Cette compétition reste cependant la référence en la matière, raison pour laquelle nous la prenons pour référence.

5.4.2 Protocole expérimental

Pour évaluer notre agent, nous utilisons le *framework* GENIUS [Lin *et al.*, 2014]. GENIUS est utilisé pour créer des sessions de négociation et des tournois. Chaque session ou tournoi utilise un domaine de négociation qui est associé à un attribut ou plus. Ces attributs peuvent être entiers ou discrets. Bien que l'introduction d'attributs continus soit prévu à terme, il semble que la version actuelle de GENIUS ne la supporte pas entièrement. En tout état de cause, les agents auxquels nous nous comparons ne sont pas conçus pour de tels attributs. Nous utilisons en effet le domaine de négociation de l'ANAC 2014 décrit par [Fujita *et al.*, 2016]. Le protocole de *bargaining* standard. Les attributs sont entiers, et varient de 0 à 10. Plusieurs variantes ont été proposées allant de 10 attributs à 50 attributs. Nous nous focalisons sur le contexte à 10 attributs. Pour chacun des contextes proposés, GENIUS fournis deux profils de préférence conflictuels, mais dont les profils de préférences forment des domaines mutuellement profitables (*e.g.*

des domaines où les deux agents obtiennent une haute utilité).

Nous moyennons les résultats sur 10 sessions, chaque agent recevant un profil de préférence puis l'autre. Nous fixons en outre la borne à 22 tours. Nous calibrons notre modèle de manière empirique. Lorsque notre agent peut choisir parmi 200 propositions, il s'en trouve généralement une qui procure à la fois une haute utilité pour lui et son adversaire. Afin que l'arbre soit également exploré en profondeur, nous lui laissons assez de temps pour qu'il génère en moyenne 50.000 simulations, c'est à dire 3 minutes. Nous avons donc, en reprenant l'Équation (5.27) :

$$\begin{aligned} 50.000^{\alpha} &= 200 \\ \alpha &= \log_{50.000}(200) = 0.489 \end{aligned} \tag{5.30}$$

5.4.3 Résultats

Nous séparons deux contextes de négociation : un contexte où la borne est présente et connue des adversaires, et un où elle est assez loin pour être considérée comme inexistante.

Adversaire	Score adversaire	Score MoCaNA	Taux d'accord
AgentM	0.882 (±0.069)	0.655 (±0.076)	0.35
DoNA	1.000 (±0.000)	0.572 (±0.000)	0.05
Gangster	0.814 (±0.130)	0.637 (±0.142)	0.45
Whale	0.779 (±0.164)	0.697 (±0.113)	0.45
Group2	**0.689** (±0.211)	**0.822** (±0.194)	0.45
kGAgent	0.977 (±0.085)	0.554 (±0.047)	0.20
AgentYk	0.806 (±0.174)	0.752 (±0.154)	0.15
BraveCat	**0.622**(±0.170)	**0.749** (±0.101)	0.95

TABLE 5.2 – Résultats de MoCaNA face aux finalistes de l'ANAC 2014

Adversaire	Score adversaire	Score MoCaNA	Taux d'accord
AgentM	0.699 (±0.078)	0.847 (±0.016)	0.15
DoNA	N/A	N/A	0
Gangster	0.534 (±0.193)	0.852 (±0.049)	0.55
Whale	0.708 (±0.064)	0.832 (±0.020)	0.45
Group2	0.530 (±0.061)	0.870 (±0.039)	0.45
kGAgent	N/A	N/A	0
AgentYk	0.712(±0.074)	0.841(±0.025)	0.10
BraveCat	0.487(±0.195)	0.855 (±0.066)	0.95

TABLE 5.3 – Résultats de MoCaNA avec élagage fixe face aux finalistes de l'ANAC 2014

5.4.3.1 ANAC 2014

Tableau 5.2 montre les résultats de la version sans élagage de notre agent – que nous avons présentée dans la Section 5.3.2.1 – face aux finalistes de l'ANAC 2014, décrits par [Fukuta *et al.*, 2016], ordonnés selon leur résultats à l'ANAC 2014 dans la catégorie utilité individuelle. Les scores représentent l'utilité moyenne de notre agent et de son adversaire moyenné sur 20 négociations, 10 avec chaque profil. Plus le score est haut, plus l'utilité de l'agent est élevée et mieux l'agent a réussi la négociation. Le nombre suivant le score précédé du signe ± est l'écart type de la série. L'utilité moyenne est calculée en ne prenant en compte que les négociations qui ont réussi La dernière colonne représente le taux d'accord atteint par les agents, c'est à dire le nombre de fois où les agents parviennent à un accord divisé par le nombre total de négociations.

Le taux d'accord très faible peut être expliqué par notre stratégie, qui ne s'accorde pas du tout à la distance à la borne, puisqu'elle ne la prend pas en compte. Cette caractéristique l'empêche de ressentir la pression du temps (en anglais *time pressure*) qui est utilisée par les autres agents pour décider s'ils concèdent beaucoup ou non. Plus de 50% des négociations se soldent par un échec. Notons que notre agent obtient de bons résultats avec l'agent avec lequel il arrive presque toujours à un accord (BraveCat). Certains agents, comme DoNA, refusent cependant de faire le moindre effort, amenant une utilité particulièrement faible à notre agent en même temps qu'un taux d'accord extrêmement bas.

Notre agent obtient de bons résultats face à BraveCat et à GROUP2. GROUP2 s'appuie sur une variante gloutonne de l'algorithme de *Coordinate Descent* et un modèle d'adversaire basé sur les

Adversaire	Score adversaire	Score MoCaNA	Taux d'accord
AgentM	0.770 (±0.113)	0.656 (±0.077)	0.10
DoNA	0.857 (±0.005)	0.547 (±0.058)	0.10
Gangster	0.563 (±0.221)	0.702 (±0.162)	0.55
Whale	0.639 (±0.085)	0.768 (±0.097)	0.25
Group2	0.534 (±0.075)	0.786 (±0.098)	0.45
kGAgent	1 (±0.0)	0.485 (±0.0)	0.05
AgentYk	0.771(±0.128)	0.637(±0.106)	0.15
BraveCat	0.519(±0.232)	0.662 (±0.121)	1.0

TABLE 5.4 – Résultats de MoCaNA avec élagage variable face aux finalistes de l'ANAC 2014

fréquences des valeurs qui sont proposées par l'adversaire. Il essaie d'optimiser sa propre utilité en début de négociation et celle des deux agents à la fin. BraveCat est basé sur un mélange de stratégies fixes, de stratégie aléatoire et de stratégie basée sur le comportement adverse (*tit-for-tat*). Il concède davantage dans les derniers tours. Ces deux agents ont un élément en commun : ils utilisent leur connaissance des bornes pour faire davantage de concessions à la fin de la négociation.

Les deux versions de l'élagage amènent une amélioration significative. La Tableau 5.3 montre le résultat de l'implémentation avec élagage fixe lorsque notre agent négocie avec les mêmes adversaires. Étant donné l'élagage pratiqué, notre agent ne fait et n'accepte que des propositions pour lesquels il recevrait une utilité de 0.8 au moins. Chaque négociation se termine donc soit sans accord, soit par un accord où notre agent obtient une utilité de 0.8 au moins. Cela a un effet important sur le résultat des négociations. Les négociations qui se terminent permettent à notre d'agent d'obtenir une bonne utilité, mais elles sont moins nombreuses que dans le cas précédent, en particulier en ce qui concerne les agents avec lesquels notre agent avait déjà un taux d'accord faible. On peut supposer que ces agents sont peu enclins à faire des concessions. Le fait que notre agent ne fasse aucune concession dont l'utilité soit inférieure à 0.8 pour lui indique que les concessions faites à l'adversaire sont également moindres, ce qui peut s'observer dans l'utilité moyenne de ces derniers. Dans ce contexte, les agents ayant une ligne de négociation « dure » qui préfèrent ne pas arriver à un accord que faire beaucoup de concessions échouent d'autant plus souvent à trouver un accord. Les autres agents voient leur utilité réduite, mais cela ne s'accompagne pas de réduction du taux d'accord.

L'autre méthode d'élagage est moins dure pour les adversaires, puisque l'élagage repose sur les

propositions qu'ils font tout au long de la négociation. Le taux d'accord n'est pas sensiblement différents du cas sans élagage. L'utilité de l'agent, en revanche, est bien meilleure que dans ce premier cas. Elle l'est moins, en revanche, que dans le cas avec un élagage fixe de 0.8. Notre agent est toujours battu par les compétiteurs ayant le mieux réussi la compétition (AgentM et DoNA) ainsi que kGAgent, dans la seule négociation ayant abouti. Il bat en revanche les autres agents. L'élagage lui permet de prendre en compte un seuil acceptable par l'adversaire, et donc meilleur que dans le cas d'un seuil fixe, tout en élaguant toutes les branches dont l'utilité d'une proposition est pire que la meilleure proposition de l'adversaire. Il ne fait donc plus aucune proposition pour laquelle son utilité est particulièrement faible, comme cela pouvait être le cas dans le cas sans élagage.

Quelle que soit la borne, elle permet aux agents de ne pas explorer les branches inintéressantes, et donc de rechercher les meilleures solutions parmi les solutions acceptables. Elles sont en cela meilleure que la pondération *a posteriori*.

Les autres agents obtiennent une utilité plus haute, mais il est assez compliqué d'en tirer des conclusions : plus de 50% des négociations échouent parce que la borne est atteinte. Il serait intéressant d'évaluer notre agent contre eux dans un contexte où la négociation n'est pas bornée ou dans un contexte où la borne est privée. Mais les agents de l'ANAC s'appuient sur cette borne pour construire leur stratégie et il est donc impossible de faire de telles expériences.

5.4.3.2 Contexte sans borne

En ce qui concerne la négociation avec le *Random Walker*, MoCaNA semble beaucoup mieux s'en sortir, même sans élagage. Il obtient un score élevé, $0.703(\pm 0.062)$ contrairement au *Random Walker*, qui obtient un score moyen de $0.378(\pm 0.077)$. ce qui confirme que notre agent est gêné par la présence de la borne, et le fait qu'il ne la connaisse pas.

Notre agent est donc un bon négociateur dès lors qu'on ne lui impose pas de borne. La présence de cette dernière coupe la négociation avant qu'il ne puisse arriver à un accord avantageux pour lui. Les négociation réussissant (moins de 50%), sont donc celles pour lesquelles il a fait le plus de concessions, ce qui a pour effet de lui conférer une utilité moindre que celle de son adversaire.

5.4.4 Discussion

L'implémentation de MoCaNA en fait un agent polyvalent également sur le plan technique : il est capable de négocier à l'intérieur de GENIUS mais aussi dans un autre environnement pour peu que ce dernier implémente les interfaces fournies par l'agent en ce qui concerne les fonctions d'utilité, le protocole et les attributs. Notre implémentation est modulaire, et utilise les ressources calculatoires qui sont à sa disposition en distribuant ses calculs. Cela lui permet de générer une proposition dans un temps acceptable, et ce malgré les besoins de calculs importants qu'engendrent les simulations de MCTS.

En comparant notre agent à ceux de l'ANAC, nous avons confirmé l'hypothèse que nous avions formulée dans la section précédente : la connaissance de la borne permet aux agents qui l'utilisent de l'exploiter aux dépens de MoCaNA. Peu de négociations s'achèvent et il s'agit généralement de celles où notre agent fait le plus de concessions. Il n'en reste pas moins qu'en plaçant notre agent dans un contexte où la borne est inconnue et assez grande pour être considérée comme inexistante, notre agent se montre performant.

5.5 Conclusion

Le but premier de ce chapitre était d'explorer les méthodes de Monte-Carlo dans le cadre de la négociation automatique en concevant un agent dont la stratégie de proposition se baserait sur cette méthode. Pour ce faire, nous avons exploité les liens entre théorie des jeux et négociation que nous avions décrits au Chapitre 4. Nous avons montré en quoi la négociation constitue un jeu particulier, et ce que cela implique du point de vue des techniques de Monte-Carlo. Nous avons ensuite décrit une architecture d'agent basée sur MCTS, et qui exploite des techniques de modélisation permettant d'inférer la stratégie de l'adversaire et son utilité. Nous avons étendu ces techniques de modélisation au cas des variables catégorielles afin que notre agent soit capable de négocier sur tous les domaines de négociation, quelle que soit la nature des attributs qui les composent. Nous avons aussi montré comment l'implémentation de notre agent est elle aussi tournée vers la polyvalence, proposant un certain nombre d'interfaces à implémenter. Enfin, notre agent est conçu pour exploiter le parallélisme, et nous avons proposé d'utiliser la parallélisation et le *multithreading* pour améliorer le temps de calcul pour générer une offre.

En faisant négocier notre agent dans un contexte non favorable, puis en mettant dans un contexte plus neutre, nous avons montré que le prix de sa polyvalence est une performance moindre face à des agents capables d'exploiter des informations sur la borne. Notre agent, qui est conçu pour négocier dans des contextes où cette borne n'existe pas forcément, perd les négociations dans ce cas. Ce résultat est à mettre en relation avec le faible taux de succès des négociations. Il semble que ces dernières ne réussissent que lorsque notre agent fait des concessions, ses adversaires refusant d'en faire eux-mêmes. Cela est confirmé par les négociations menées dans le cas plus général où nous avons fait disparaître virtuellement la borne. En effet, notre agent se montre beaucoup plus performant dans ce cas.

Au-delà de notre agent lui-même, nous espérons que ces travaux permettront de jeter un pont entre les techniques d'IA pour les jeux et les techniques de négociation automatique. Bien que la négociation automatique constitue un jeu très particulier, elle n'en reste pas moins un jeu extensif. En ce qui concerne notre agent, il serait intéressant de le placer dans un contexte plus ouvert que celui de l'ANAC, par exemple dans un protocole m:n, tel que ceux décrits par [An *et al.*, 2009, An *et al.*, 2016].

Notre agent est théoriquement capable de négocier dans ce contexte. Il serait aussi intéressant de l'étendre afin d'intégrer des modifications telles que RAVE ou AMAF. Il y a cependant un bémol à ces améliorations. S'il y a une borne connue à la négociation et que les agents s'appuient dessus, alors les scores calculés par AMAF seront possiblement biaisés par la modélisation de la stratégie.

Conclusion

1 Introduction

Dans cette thèse, nous nous sommes intéressés à la description d'une place de marché intelligente dédiée à l'achat et à la vente de créance. Cette place de marché est utilisée par des fournisseurs souhaitant réduire leur besoin en fonds de roulement en le faisant financer par des banques et des fonds. Il faut rappeler que cette thèse se place dans un cadre appliqué, celui d'un Fonds Unique Interministériel dont le but est la gestion du besoin en fonds de roulement. Notre plate-forme s'adresse donc à un public bien particulier : des investisseurs pourvus de moyens importants d'une part et des fournisseurs qui sont souvent des PME, et manquent de fonds d'autres part. Cette asymétrie est à prendre en compte dans la conception de notre plate-forme. Nous ne souhaitons pas que les acteurs les mieux pourvus financièrement puissent l'utiliser aux dépens des PME dont les moyens sont réduits. À cette fin, il est nécessaire de faire en sorte que le protocole proposé par la plate-forme n'avantage pas les acteurs les plus riches en leur permettant de fournir à des algorithmes d'apprentissage des données qu'ils auraient acquises gratuitement, sans même participer au financement des fournisseurs. Il est aussi nécessaire de mettre à disposition des fournisseurs un agent de négociation qui soit capable de négocier quelles que soient les contraintes qui lui sont imposées.

2 Contributions

Le première étape de la réalisation de la plate-forme a consisté à décrire la structure et les besoins soulevés par notre place de marché. Nous avons pour cela conçu OntoMASK, une ontologie qui nous a permis de détailler les différents concepts de la plate-forme et les relations qu'ils entretiennent.

Selon [Baarslag *et al.*, 2015b], la négociation peut être décrite par trois éléments : son protocole, son objet et les agents qui négocient. Les biens étant déjà définis par la problématique, nous nous sommes principalement concentrés sur le protocole utilisé par la place de marché et sur les agents de négociation. Nous avons soulevé la problématique posée par la gestion des agents curieux dans le protocole de *bargaining*, dont la cause réside dans l'automatisation de la négociation. Ces agents utilisent la place de marché pour obtenir des informations sur leurs adversaires sans nécessairement chercher à obtenir un bien. Ces informations alimentent ultérieurement des algorithmes d'apprentissage automatique dans le cadre d'une modélisation de l'adversaire qui peut déséquilibrer la négociation. Leur comportement génère à cette fin des négociations inutiles se terminant par un échec. Ils causent aussi un allongement artificiel des négociations. Nous avons conçu un protocole qui leur donne une incitation à agir comme des agents incurieux.

Nous avons également conçu un agent de négociation automatique qui explore les possibilités offertes par les méthodes d'apprentissage automatique et par les méthodes d'intelligence artificielle pour les jeux comme *Monte Carlo Tree Search*. Cet agent est capable de négocier dans un grand spectre de situations, incluant la compétition de négociation automatique qui a servi de base au développement de nouvelles techniques mais aussi dans des situations plus concrètes où les attributs négociés sont continus et la négociation non limitée par le protocole.

2.1 Agents curieux

L'essor de la négociation automatique et des intelligences artificielles développées dans ce cadre a changé la manière d'aborder les négociateurs, les protocoles et les objets de négociation. Si ce derniers n'ont pas profondément changé – ils bénéficient simplement d'une formalisation plus précise – les autres aspects ont eux radicalement évolué : les agents de négociation automatique sont capables de s'envoyer des milliers de propositions sans jamais se fatiguer, de modéliser mathématiquement la stratégie ou le prix de réserve de leur adversaire, et de faire varier leur stratégie en fonction du contexte.

Si cette évolution est globalement positive, elle soulève les mêmes interrogations que celles sur l'intelligence artificielle en général, surtout lorsque cela permet d'inférer un certain nombre d'informations sur les autres acteurs négociant, y compris des informations privées. Il il faut aussi se demander dans quelle mesure nous pouvons empêcher les débordements d'agents dont le but serait précisément l'obtention d'informations privées. En formalisant la négociation à l'aide de la théorie des jeux, il est possible de trouver des moyens de contrer de tels comportements.

Pour lutter contre ces agents, nous avons proposé un protocole de négociation bilatérale basé sur le *bargaining*, mais qui donne une incitation aux agents à ne pas rompre la négociation. Cette solution se base sur la formalisation de la négociation comme un jeu extensif, et l'implémentation d'un mécanisme au sens du *mechanism design*. Notre protocole modifie le *bargaining* de trois façons : d'abord, il demande aux agents de révéler leur prix de réserve de manière sécurisée. Ensuite, il limite le nombre de propositions échangées et enfin, il force l'échange au prix de réserve annoncé si les agents n'arrivent pas à se mettre d'accord.

Nous avons démontré un certain nombre de propriétés sur le mécanisme ainsi défini : une incitation partielle à l'honnêteté, une incitation à chercher un accord entre les prix déclarés. Nous avons également démontré qu'il permet aux agents « discrets » qui souhaitent garder secrètes leurs informations privées d'améliorer leur utilité, alors qu'il diminuait l'utilité des agents curieux. Notre protocole n'est cependant pas tout à fait stable. En effet, du fait que le prix au pire est fonction du prix de réserve déclaré par l'agent, celui-ci peut tenter d'augmenter ses attentes en espérant ainsi gagner davantage au cours du *bargaining*. Nous avons cependant expliqué comment il était possible de réduire cette tentation en diminuant les probabilités que la négociation ne démarre si les prix de réserve sont proches.

Nous avons également proposé une extension de ce protocole au cas multi-attributs, en utilisant des ensembles d'offres acceptables, et avons proposé trois implémentations, une centralisée et une décentralisée pour la négociation 1:1 et une troisième implémentation pour le cas 1:n.

2.2 MoCaNa

En utilisant la formalisation de la négociation comme jeu extensif, nous avons exploré les possibilités qu'offrent les méthodes utilisées en intelligence artificielle pour les jeux, et plus particulièrement les

méthodes de Monte-Carlo.

Cependant, nous avons également montré que le *bargaining* n'est pas un jeu extensif ordinaire, et que ses particularités forcent à adapter l'algorithme traditionnel de MCTS. En particulier, il est impossible d'utiliser les *Upper Confidence Trees*. Des méthodes adaptées au contexte continu, comme le *progressive widening* semblent plus pertinentes. La nature de la négociation – un jeu extensif – pousse aussi à chercher à connaître l'équivalent de son type, c'est à dire sa fonction d'utilité. Il est donc nécessaire de modéliser l'adversaire pour savoir ce qu'il cherche, et ainsi trouver la meilleure offre possible acceptable pour lui. Nous avons proposé d'utiliser deux méthodes : la régression de processus gaussien pour apprendre sa stratégie, et l'apprentissage bayésien pour apprendre son utilité. Ces techniques ont traditionnellement été utilisées dans des domaines de négociation continu. Nous avons proposé de les étendre au contexte discret.

Nous avons ensuite présenté notre implémentation de ces principes sous la forme d'un agent de négociation automatique nommé MoCaNA pour *Monte Carlo-based Negotiation Agent*. Notre implémentation est multithreadée et découplée de tout framework, mais nous avons également développé une interface avec GENIUS, le *framework* de la Compétition de Négociation Automatique pour Agents (ANAC).

Nous avons ensuite testé notre agent face aux agents de l'ANAC. Cependant, ceux-ci sont conçu pour négocier dans un contexte particulier, où la négociation est limitée soit en temps, soit en nombre de tours, et où les attributs sont des entiers. Notre agent n'est pas spécifiquement conçu pour ce contexte, et n'exploite donc pas ces informations, contrairement aux autres agents qui les exploitent à ses dépens. Cela explique la proportion importante de négociations n'aboutissant pas, de même que le fait qu'il est systématiquement à l'origine des concessions quand elles aboutissent. Cette intuition est confirmée par la négociation contre un agent aléatoire ne connaissant pas non plus la borne, et ne prenant pas en compte les informations sur la nature des attributs du domaine de négociation.

3 Perspectives

Un élément apparaît en filigrane dans l'ensemble de cette thèse : l'assimilation de la négociation automatique à un jeu extensif à information incomplète. Nous espérons qu'au-delà des résultats cités plus haut, ces travaux ouvriront la voie à un rapprochement entre négociation automatique et théorie des jeux. Ce rapprochement, qui avait notamment été amorcé par [Nash, 1950, Myerson, 1979] et poursuivi par [Rubinstein, 1982] semble avoir été écarté avec l'avènement de modèles plus complexes. Il est surprenant que les travaux menés sur les jeux, qu'il s'agisse de travaux théoriques sur la théorie des jeux ou de travaux sur l'intelligence artificielle pour les jeux, aient autant divergé de ceux concernant la négociation automatique.

3.1 Agents curieux

Outre cette perspective générale, il manque à la partie protocole de notre travail une incitation totale à l'honnêteté. Cette incitation est complexe à mettre en œuvre parce que le prix payé par l'agent au pire dépend nécessairement de son prix de réserve. Mais en travaillant dans le contexte multilatéral, un protocole permettant à la fois de conserver les propriétés de notre protocole et d'inciter à l'honnêteté pourrait voir le jour.

Une autre perspective de ce travail est le relâchement des contraintes sur le domaine de négociation. Nous avons amorcé le premier pas dans cette direction en étendant notre modèle du prix au cas multi-attributs continus. Nous avons cependant mis un certain nombre de contraintes sur l'utilité des agents dans ce cas, afin que les propriétés du protocole restent valable. Il serait intéressant de relâcher ces contraintes, et de démontrer quelles sont les conditions minimales sur l'utilité pour que ces propriétés restent valables. Il serait intéressant de voir comment ce protocole peut être adapté aux attributs numériques et catégoriels, ainsi qu'aux domaines de négociation mixtes.

Enfin, nous n'avons pas eu l'occasion de voir l'effet de notre protocole à l'œuvre sur la plate-forme. En particulier, nous n'avons pas pu observer son effet sur la curiosité de sujets humains. Une telle expérience est loin d'être superflue. L'expérience montre assez à quel point la rationalité humaine diffère de celle de la machine, et à quel point nous pouvons être imprévisibles. Une idée serait par exemple de reproduire les expériences que nous avons menées sur des agents artificiels et de les transposer, si possible à des acteurs habitués à de telles négociations.

3.2 MoCaNA

Nous avons implémenté un algorithme de Monte Carlo simple. Or, il existe de très nombreuses améliorations pour cet algorithme. Il serait intéressant de voir comment ces différentes modifications altèrent les résultats de notre agent.

Il serait aussi intéressant de voir dans quelle mesure les différents modules interagissent. Tester la stratégie d'offre avec d'autres algorithmes de modélisation d'adversaires, peut-être plus spécialisés, pourrait permettre à MoCaNA de prendre en compte les éléments fournis par le protocole de l'ANAC, et de négocier à armes égales avec les autres agents.

Nous avons conservé une supposition sur les attributs du domaine de négociation, à savoir leur indépendance. Pour que notre agent soit vraiment capable de négocier dans toutes les situations, il faudrait trouver des méthodes de modélisation qui restent aussi générales que celles que nous avons proposées tout en ne faisant pas cette supposition.

D'un point de vue implémentation, nous pourrions étendre notre parallélisation et implémenter un agent qui soit capable d'utiliser le GPU, permettant ainsi de faire un très grand nombre de simulations simultanées et d'accélérer le temps pris pour faire une proposition.

Enfin, il serait intéressant de placer notre agent dans un contexte différent, de l'appliquer à d'autres jeux de négociation. Nous avons notamment conçu notre agent pour qu'il soit utilisable dans un contexte m:n tel que celui décrit par [An *et al.*, 2016].

Bibliographie

[eba, 1995] (1995). Site internet d'ebay. `http://www.ebay.com`.

[ANA, 2011] (2011). The automated negociation agent competition website. `http://mmi.tudelft.nl/negotiation/index.php/Automated_Negotiating_Agents_Competition_(ANAC)`.

[Jav, 2013] (2013). Nist java numerics website. `http://math.nist.gov/javanumerics/`.

[Oct, 2017] (2017). The gnu octave website. `https://www.gnu.org/software/octave/`.

[Agrawal et Mishra, 2012] Agrawal, M. et Mishra, P. (2012). A comparative survey on symmetric key encryption techniques. *International Journal on Computer Science and Engineering*, 4(5):877.

[An *et al.*, 2016] An, B., Gatti, N. et Lesser, V. (2016). Alternating-offers bargaining in one-to-many and many-to-many settings. *Annals of Mathematics and Artificial Intelligence*, 77(1):67–103.

[An *et al.*, 2009] An, B., Gatti, N. et Lesser, V. R. (2009). Extending alternating-offers bargaining in one-to-many and many-to-many settings. *In International Joint Conferences on Web Intelligence and Intelligent Agent Technologies*, volume 2, pages 423–426.

[An et Lesser, 2012] An, B. et Lesser, V. (2012). Yushu : A heuristic-based agent for automated negotiating competition. *In* Ito, T., Zhang, M., Robu, V., Fatima, S. et Matsuo, T., éditeurs : *New Trends in Agent-Based Complex Automated Negotiations*, pages 145–149. Springer Berlin Heidelberg, Berlin, Heidelberg.

[An *et al.*, 2010] An, B., Lesser, V., Irwin, D. et Zink, M. (2010). Automated negotiation with decommitment for dynamic resource allocation in cloud computing. *In Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems : Volume 1 - Volume 1*, AAMAS '10, pages 981–988, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Anderson, 2003] Anderson, T. W. (2003). *An introduction to multivariate statistical analysis*. Wiley New York, 3rd édition.

[Arrow et Debreu, 1954] Arrow, K. et Debreu, G. (1954). Existence of an equilibrium for a competitive economy. *Econometrica*, 22(3):265–290.

[Arrow, 1951] Arrow, K. J. (1951). An extension of the basic theorems of classical welfare economics. *In Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 507–532, Berkeley, Calif. University of California Press.

[Arrow et Hurwicz, 1958] Arrow, K. J. et Hurwicz, L. (1958). On the stability of the competitive equilibrium, i. *Econometrica*, 26(4):522–552.

[Atradius, 2017] Atradius (2017). Baromètre atradius sur les pratiques de paiement. Rapport.

[Aydoğan et Yolum, 2012] Aydoğan, R. et Yolum, P. (2012). Learning opponent's preferences for effective negotiation : an approach based on concept learning. *Autonomous Agents and Multi-Agent Systems*, 24(1):104–140.

[Baarslag, 2016] Baarslag, T. (2016). *Exploring the Strategy Space of Negotiating Agents : A Framework for Bidding, Learning and Accepting in Automated Negotiation*, chapitre A Component-Based Architecture to Explore the Space of Negotiation Strategie, pages 53–69. Springer International Publishing, Cham.

[Baarslag *et al.*, 2015a] Baarslag, T., Aydoğan, R., Hindriks, K. V., Fujita, K., Ito, T. et Jonker, C. M. (2015a). The automated negotiating agents competition, 2010–2015. *AI Magazine*, 34(6):115–118.

[Baarslag *et al.*, 2015b] Baarslag, T., Hendrikx, M. J. C., Hindriks, K. V. et Jonker, C. M. (2015b). Learning about the opponent in automated bilateral negotiation : a comprehensive survey of opponent modeling techniques. *Autonomous Agents and Multi-Agent Systems*, 20(1):1–50.

[Baarslag *et al.*, 2013] Baarslag, T., Hindriks, K. et Jonker, C. (2013). A tit for tat negotiation strategy for real-time bilateral negotiation. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, volume 435, pages 229–233. Springer Berlin Heidelberg.

[Baarslag *et al.*, 2014] Baarslag, T., Hindriks, K. et Jonker, C. (2014). Effective acceptance conditions in real-time automated negotiation. *Decision Support Systems*, 60(Supplement C):68 – 77. Automated Negotiation Technologies and their Applications.

[Baarslag et Hindriks, 2013] Baarslag, T. et Hindriks, K. V. (2013). Accepting optimally in automated negotiation with incomplete information. *In Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, pages 715–722, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Bauer *et al.*, 2001] Bauer, B., Müller, J. P. et Odell, J. (2001). Agent uml : A formalism for specifying multiagent software systems. *International Journal of Software Engineering and Knowledge Engineering*, 11(03):207–230.

[Belleili *et al.*, 2004] Belleili, H., Bouzid, M. et Sellami, M. (2004). A fast negotiation algorithm for multiple methods agents. *In Proceedings. 2004 International Conference on Information and Communication Technologies : From Theory to Applications, 2004.*, pages 495–496.

[Benyoucef *et al.*, 2000] Benyoucef, M., Keller, R. K., Lamouroux, S., Robert, J. et Trussart, V. (2000). Towards a generic e-negotiation platform. *In Sixth Int. Conference on Re-Technologies for Information Systems, Austrian Computer Society*, pages 95–109.

[Blume et Durlauf, 2008] Blume, L. E. et Durlauf, S. F., éditeurs (2008). *The New Palgrave Dictionary of Economics*. Palgrave Macmillan.

[Bowman, 2012] Bowman, F. (2012). *Introduction to Bessel functions*. Courier Corporation.

[Brams, 2003] Brams, S. J. (2003). *Negotiation Games : Applying game theory to bargaining and arbitration*, volume 2. Psychology Press.

[Brams et Taylor, 1996] Brams, S. J. et Taylor, A. D. (1996). *Fair Division : From cake-cutting to dispute resolution*. Cambridge University Press.

[Browne *et al.*, 2012] Browne, C. C., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Taverner, S., Perez, D., Samothrakis, S. et Colton, S. (2012). A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

[Buffett *et al.*, 2006] Buffett, S., Comeau, L., Spencer, B. et Fleming, M. W. (2006). Detecting opponent concessions in multi-issue automated negotiation. *In Proceedings of the 8th International Conference on Electronic Commerce : The New e-Commerce : Innovations for Conquering Current Barriers, Obstacles and Limitations to Conducting Successful Business on the Internet*, ICEC '06, pages 11–18, New York, NY, USA. ACM.

[Carbonneau *et al.*, 2008] Carbonneau, R., Kersten, G. E. et Vahidov, R. (2008). Predicting opponent's moves in electronic negotiations using neural networks. *Expert Systems with Applications*, 34(2):1266–1273.

[Cazenave, 2006] Cazenave, T. (2006). A phantom-go program. *In* van den Herik, H. J., Hsu, S.-C., Hsu, T.-s. et Donkers, H. H. L. M. J., éditeurs : *Advances in Computer Games : 11th International Conference, ACG 2005, Taipei, Taiwan, September 6-9, 2005. Revised Papers*, pages 120–125, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Chaslot *et al.*, 2008a] Chaslot, G., Bakkes, S., Szita, I. et Spronck, P. (2008a). Monte-carlo tree search : A new framework for game ai. *In Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference*, pages 216–217. AAAI Press.

[Chaslot *et al.*, 2008b] Chaslot, G., Winands, M. H. M. et van den Herik, H. J. (2008b). Parallel Monte Carlo tree search. *In* van den Herik, H. J., Xu, X., Ma, Z. et Winands, M. H. M., éditeurs : *Computers and Games : 6th International Conference, CG 2008, Beijing, China, September 29 - October 1, 2008. Proceedings*, pages 60–71, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Chaum *et al.*, 1988] Chaum, D., Crépeau, C. et Damgard, I. (1988). Multiparty unconditionally secure protocols. *In Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, pages 11–19, New York, NY, USA. ACM.

[Chavez et Maes, 1996] Chavez, A. et Maes, P. (1996). Kasbah : An agent marketplace for buying and selling goods. *In Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, volume 31, pages 75–90.

[Chen *et al.*, 2008] Chen, D.-N., Jeng, B., Lee, W.-P. et Chuang, C.-H. (2008). An agent-based model for consumer-to-business electronic commerce. *Expert Systems with Applications*, 34(1):469 – 481.

[Chen et Weiss, 2012] Chen, S. et Weiss, G. (2012). A novel strategy for efficient negotiation in complex environments. *In* Timm, I. J. et Guttmann, C., éditeurs : *Multiagent System Technologies : 10th German Conference Proceedings*, pages 68–82. Springer Berlin Heidelberg.

[Chen et Weiss, 2013] Chen, S. et Weiss, G. (2013). An efficient automated negotiation strategy for complex environments. *Engineering Applications of Artificial Intelligence*, 26(10):2613–2623.

[Chen et Weiss, 2014] Chen, S. et Weiss, G. (2014). Omac : A discrete wavelet transformation based negotiation agent. *In* Marsa-Maestre, I., Lopez-Carmona, M. A., Ito, T., Zhang, M., Bai, Q. et Fujita, K., éditeurs : *Novel Insights in Agent-based Complex Automated Negotiation*, pages 187–196. Springer Japan, Tokyo.

[Chen et Hu, 2011] Chen, X. et Hu, C. (2011). The value of supply chain finance. *In* Habib, M., éditeur : *Supply Chain Management - Applications and Simulations*, pages 111–132. INTECH.

[Cheng et Wellman, 1998] Cheng, J. Q. et Wellman, M. P. (1998). The walras algorithm : A convergent distributed implementation of general equilibrium outcomes. *Computational Economics*, 12(1):1–24.

[Choi *et al.*, 2009] Choi, H.-L., Brunet, L. et How, J. P. (2009). Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 25(4):912–926.

[Couëtoux, 2013] Couëtoux, A. (2013). *Monte Carlo Tree Search for Continuous and Stochastic Sequential Decision Making Problems*. Thèse de doctorat, Université Paris 11.

[Cramton *et al.*, 2006] Cramton, P., Shoham, Y. et Steinberfg, R., éditeurs (2006). *Combinatorial Auctions*. MIT Press.

[Dasgupta *et al.*, 1999] Dasgupta, P., Narasimhan, N., Moser, L. E. et Melliar-Smith, P. M. (1999). Magnet : mobile agents for networked electronic trading. *IEEE Transactions on Knowledge and Data Engineering*, 11(4):509–525.

[Davis et Smith, 1983] Davis, R. et Smith, R. G. (1983). Negotiation as a metaphor for distributed problem solving. *Artificial intelligence*, 20(1):63–109.

[de Jonge et Sierra, 2016] de Jonge, D. et Sierra, C. (2016). Gangster : An automated negotiator applying genetic algorithms. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 225–234. Springer International Publishing.

[de Jonge et Zhang, 2017] de Jonge, D. et Zhang, D. (2017). Automated negotiations for general game playing. *In Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '17, pages 371–379, Richland, SC.

[de Vries et Vohra, 2003] de Vries, S. et Vohra, R. V. (2003). Combinatorial auctions : A survey. *INFORMS Journal on Computing*, 15(3):284–309.

[Debreu, 1974] Debreu, G. (1974). Excess demand functions. *Journal of Mathematical Economics*, 1(1):15 – 21.

[Di Napoli *et al.*, 1996] Di Napoli, C., Sierra, C., Giordano, M., Noriega, P. et Furnari, M. M. (1996). A pvm implementation of the fishmarket multiagent system. *In Proceedings Mexico-USA Collaboration in Intelligent Systems Technologies.*, pages 68–76.

[Dirkzwager et Hendrikx, 2014] Dirkzwager, A. et Hendrikx, M. (2014). An adaptive negotiation strategy for real-time bilateral negotiations. *In* Marsa-Maestre, I., Lopez-Carmona, M. A., Ito, T., Zhang, M., Bai, Q. et Fujita, K., éditeurs : *Novel Insights in Agent-based Complex Automated Negotiation*, pages 163–170. Springer Japan, Tokyo.

[Dirkzwager *et al.*, 2013] Dirkzwager, A. S. Y., Hendrikx, M. J. C. et De Ruiter, J. R. (2013). Thenegotiator : A dynamic strategy for bilateral negotiations with time-based discounts. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, pages 217–221. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Domingos, 2012] Domingos, P. (2012). A few useful things to know about machine learning. *Communications of ACM*, 55(10):78–87.

[Doorenbos *et al.*, 1997] Doorenbos, R. B., Etzioni, O. et Weld, D. S. (1997). A scalable comparison-shopping agent for the world-wide web. *In Proceedings of the First International Conference on Autonomous Agents*, AGENTS '97, pages 39–48. ACM.

[Egmont-Petersen *et al.*, 2002] Egmont-Petersen, M., de Ridder, D. et Handels, H. (2002). Image processing with neural networks—a review. *Pattern Recognition*, 35(10):2279 – 2301.

[Eiben et Smith, 2015] Eiben, A. E. et Smith, J. E. (2015). *Introduction to Evolutionary Computing*. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Erez et Zuckerman, 2016] Erez, E. S. et Zuckerman, I. (2016). Dona—a domain-based negotiation agent. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 261–271. Springer International Publishing.

[Fang *et al.*, 2008] Fang, F., Xin, Y., Xia, Y. et Haitao, X. (2008). An opponent's negotiation behavior model to facilitate buyer-seller negotiations in supply chain management. *In 2008 International Symposium on Electronic Commerce and Security*.

[Faratin *et al.*, 1998] Faratin, P., Jennings, N. R. et Sierra, C. (1998). Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3-4):159–182.

[Faratin *et al.*, 2002] Faratin, P., Sierra, C. et Jennings, N. R. (2002). Using similarity criteria to make issue trade-offs in automated negotiations. *Artificial Intelligence*, 142(2):205 – 237. International Conference on MultiAgent Systems 2000.

[Fershtman, 1990] Fershtman, C. (1990). The importance of the agenda in bargaining. *Games and Economic Behavior*, 2(3):224–238.

[Fershtman, 2000] Fershtman, C. (2000). A note on multi-issue two-sided bargaining : Bilateral procedures. *Games and Economic Behavior*, 30(2):216–227.

[Finnsson, 2012] Finnsson, H. (2012). Generalized Monte Carlo tree search extensions for general game playing. *In Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 1550–1556. AAAI Press.

[Fishel *et al.*, 2013] Fishel, R., Bercovitch, M. et Gal, Y. K. (2013). Bram agent. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, pages 213–216. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Fisher, 1976] Fisher, F. M. (1976). A non-tâtonnement model with production and consumption. *Econometrica*, 44(5):907–938.

[Fisher, 1983] Fisher, F. M. (1983). *Disequilibrium Foundations of Equilibrium Economics*. Cambridge University Press.

[Frieder et Miller, 2013] Frieder, A. et Miller, G. (2013). Value model agent : A novel preference profiler for negotiation with agents. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, pages 199–203. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Fujita *et al.*, 2016] Fujita, K., Aydoğan, R., Baarslag, T., Ito, T. et Jonker, C. (2016). The fifth automated negotiating agents competition (anac 2014). *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 211–224. Springer International Publishing.

[Fukuta *et al.*, 2016] Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs (2016). *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*. Springer International Publishing.

[Furtak et Buro, 2013] Furtak, T. et Buro, M. (2013). Recursive monte carlo search for imperfect information games. *In 2013 IEEE Conference on Computational Inteligence in Games (CIG)*, pages 1–8.

[Gibbard, 1973] Gibbard, A. (1973). Manipulation of voting schemes : A general result. *Econometrica*, 41(4):587–601.

[Gneezy *et al.*, 2003] Gneezy, U., Haruvy, E. et Roth, A. E. (2003). Bargaining under a deadline : evidence from the reverse ultimatum game. *Games and Economic Behavior*, 45(2):347–368.

[Golbreich *et al.*, 2009] Golbreich, C., Wallace, E. K. et Patel-Schneider, P. F. (2009). Owl 2 web ontology language : new features and rationale.

[Goudet *et al.*, 2015] Goudet, O., Kant, J.-D. et Ballot, G. (2015). Forbidding fixed duration contracts : Unfolding the opposing consequences with a multi-agent model of the french labor market. *In* Amblard, F., Miguel, F. J., Blanchet, A. et Gaudou, B., éditeurs : *Advances in Artificial Economics*, pages 151–167, Cham. Springer International Publishing.

[Grabisch et Labreuche, 2010] Grabisch, M. et Labreuche, C. (2010). A decade of application of the choquet and sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1):247–286.

[Groves, 1973] Groves, T. (1973). Incentives in teams. *Econometrica*, 41(4):617–631.

[Gupta et Dutta, 2011] Gupta, S. et Dutta, K. (2011). Modeling of financial supply chain. *European Journal of Operational Research*, 211(1):47–56.

[Guttman et Maes, 1999] Guttman, R. H. et Maes, P. (1999). Agent-mediated integrative negotiation for retail electronic commerce. *In* Noriega, P. et Sierra, C., éditeurs : *Agent Mediated Electronic Commerce : First International Workshop on Agent Mediated Electronic Trading AMET-98 Minneapolis, MN, USA, May 10th, 1998 Selected Papers*, pages 70–90, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Guttman *et al.*, 1998] Guttman, R. H., Moukas, A. G. et Maes, P. (1998). Agent-mediated electronic commerce : a survey. *The Knowledge Engineering Review*, 13(2):147–159.

[Hahn et Negishi, 1962] Hahn, F. H. et Negishi, T. (1962). A theorem on non-tâtonnement stability. *Econometrica*, 30(3):463–469.

[Halevy *et al.*, 2009] Halevy, A., Norvig, P. et Pereira, F. (2009). The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12.

[Hao et Leung, 2012] Hao, J. et Leung, H.-F. (2012). Abines : An adaptive bilateral negotiating strategy over multiple items. *In 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 95–102.

[Hao et Leung, 2014] Hao, J. et Leung, H.-f. (2014). Cuhkagent : An adaptive negotiation strategy for bilateral negotiations over multiple items. *In* Marsa-Maestre, I., Lopez-Carmona, M. A., Ito, T., Zhang, M., Bai, Q. et Fujita, K., éditeurs : *Novel Insights in Agent-based Complex Automated Negotiation*, pages 171–179. Springer Japan, Tokyo.

[Hazan *et al.*, 2006] Hazan, E., Safra, S. et Schwartz, O. (2006). On the complexity of approximating k-set packing. *Computational complexity*, 15(1):20–39.

[He *et al.*, 2003] He, M., Jennings, N. R. et Leung, H.-F. (2003). On agent-mediated electronic commerce. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):985–1003.

[Hindriks et Tykhonov, 2008] Hindriks, K. et Tykhonov, D. (2008). Opponent modelling in automated multi-issue negotiation using bayesian learning. *In Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 331–338. International Foundation for Autonomous Agents and Multiagent Systems.

[Hofmann, 2005] Hofmann, E. (2005). Supply chain finance-some conceptual insights. *In* Lasch, R. et Janker, C. G., éditeurs : *Logistik Management Innovative Logistikkonzepte*, pages 203–214. Deutscher Universitats-Verlag.

[Hou, 2004] Hou, C. (2004). Predicting agents tactics in automated negotiation. *In Proceedings. IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2004. (IAT 2004).*, pages 127–133.

[Huai et Sandholm, 1999] Huai, Q. et Sandholm, T. W. (1999). Mobile agents in an electronic auction house. *In Mobile Agents in the Context of Competition and Cooperation (MAC3) Workshop at the Third International Conference on Autonomous Agents (AGENTS)*, pages 24–33.

[Huang *et al.*, 1998] Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Shih, H. H., Zheng, Q., Yen, N.-C., Tung, C. C. et Liu, H. H. (1998). The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *In Proceedings of the Royal Society of London A : mathematical, physical and engineering sciences*, volume 454, pages 903–995. The Royal Society.

[Hurwicz et Reiter, 2006] Hurwicz, L. et Reiter, S. (2006). *Designing economic mechanisms*. Cambridge University Press.

[Inderst, 2000] Inderst, R. (2000). Multi-issue bargaining with endogenous agenda. *Games and Economic Behavior*, 30(1):64–82.

[Ioannidis et Grama, 2003] Ioannidis, I. et Grama, A. (2003). An efficient protocol for yao's millionaires' problem. *In 36th Annual Hawaii International Conference on System Sciences, 2003. Proceedings of the*, pages 6 pp.–.

[Jøsang *et al.*, 2007] Jøsang, A., Ismail, R. et Boyd, C. (2007). A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618 – 644. Emerging Issues in Collaborative Commerce.

[Kadono, 2016] Kadono, Y. (2016). Agent yk : An efficient estimation of opponent's intention with stepped limited concessions. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 279–283. Springer International Publishing.

[Kakimoto et Fujita, 2016] Kakimoto, S. et Fujita, K. (2016). k-gagent : Negotiating agents considering interdependencies between issues. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 241–248. Springer International Publishing.

[Karp, 1972] Karp, R. M. (1972). Reducibility among combinatorial problems. *In* Miller, R. E., Thatcher, J. W. et Bohlinger, J. D., éditeurs : *Complexity of Computer Computation*, The IBM Research Symposia Series, pages 85–103. Springer US.

[Kawaguchi *et al.*, 2012] Kawaguchi, S., Fujita, K. et Ito, T. (2012). Agentk : Compromising strategy based on estimated maximum utility for automated negotiating agents. *In* Ito, T., Zhang, M., Robu, V., Fatima, S. et Matsuo, T., éditeurs : *New Trends in Agent-Based Complex Automated Negotiations*, pages 137–144. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Kawaguchi *et al.*, 2013] Kawaguchi, S., Fujita, K. et Ito, T. (2013). Agentk2 : Compromising strategy based on estimated maximum utility for automated negotiating agents. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, pages 235–241. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Klemperer, 1999] Klemperer, P. (1999). Auction theory : A guide to the literature. *Journal of Economic Surveys*, 13(3):227–286.

[Kocsis et Szepesvári, 2006] Kocsis, L. et Szepesvári, C. (2006). Bandit based monte-carlo planning. *In* Fürnkranz, J., Scheffer, T. et Spiliopoulou, M., éditeurs : *Machine Learning : ECML 2006 : 17th European Conference on Machine Learning Berlin, Germany, September 18-22, 2006 Proceedings*, pages 282–293. Springer Berlin Heidelberg.

[Koeman *et al.*, 2015] Koeman, V. J., Boon, K., van den Oever, J. Z., Dumitru-Guzu, M. et Stanculescu, L. C. (2015). The fawkes agent—the anac 2013 negotiation contest winner. *In* Fujita, K., Ito, T., Zhang, M. et Robu, V., éditeurs : *Next Frontier in Agent-based Complex Automated Negotiation*, pages 143–151. Springer Japan, Tokyo.

[Kraus, 2001] Kraus, S. (2001). Automated negotiation and decision making in multiagent environments. *In* Luck, M., Mařík, V., Štěpánková, O. et Trappl, R., éditeurs : *Multi-Agent Systems and Applications*, volume 2086, pages 150–172. Springer Berlin Heidelberg.

[Kuhn, 1962] Kuhn, H. W. (1962). Game theory and models of negotiation. *Journal of Conflict Resolution*, 6(1):1–4.

[Kupferschmid et Helmert, 2007] Kupferschmid, S. et Helmert, M. (2007). A skat player based on monte-carlo simulation. *In* van den Herik, H. J., Ciancarini, P. et Donkers, H. H. L. M. J., éditeurs : *Computers and Games : 5th International Conference, CG 2006, Turin, Italy, May 29-31, 2006. Revised Papers*, pages 135–147, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Kurbel et Loutchko, 2003] Kurbel, K. et Loutchko, I. (2003). Towards multi-agent electronic marketplaces : what is there and what is missing ? *The Knowledge Engineering Review*, 18(1):33–46.

[Liang et Huang, 2000] Liang, T.-P. et Huang, J.-S. (2000). A framework for applying intelligent agents to support electronic trading. *Decision Support Systems*, 28(4):305 – 317.

[Lin et Tzeng, 2005] Lin, H.-Y. et Tzeng, W.-G. (2005). An efficient solution to the millionaires' problem based on homomorphic encryption. *In* Ioannidis, J., Keromytis, A. et Yung, M., éditeurs : *Applied Cryptography and Network Security*, pages 456–466, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Lin *et al.*, 2014] Lin, R., Kraus, S., Baarslag, T., Tykhonov, D., Hindriks, K. et Jonker, C. M. (2014). Genius : an integrated environment for supporting the design of generic automated negotiators. *Computational Intelligence*, 30(1):48–70.

[Lin *et al.*, 2006] Lin, R., Kraus, S., Wilkenfeld, J. et Barry, J. (2006). An automated agent for bilateral negotiation with bounded rational agents with incomplete information. *In Proc. of the 17th European Conference on Artificial Intelligence (ECAI)*, volume 141, pages 270–274.

[Luce, 2005] Luce, R. D. (2005). *Individual choice behavior : A theoretical analysis*. Courier Corporation.

[Lütkepohl, 2005] Lütkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer Science & Business Media.

[Mailler *et al.*, 2003] Mailler, R., Lesser, V. et Horling, B. (2003). Cooperative negotiation for soft real-time distributed resource allocation. *In Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '03, pages 576–583, New York, NY, USA. ACM.

[Mantel, 1974] Mantel, R. R. (1974). On the characterization of aggregate excess demand. *Journal of Economic Theory*, 7(3):348 – 353.

[Mason et Handscomb, 2002] Mason, J. C. et Handscomb, D. C. (2002). *Chebyshev polynomials*. CRC Press.

[Masvoula, 2013] Masvoula, M. (2013). Forecasting negotiation counterpart's offers : A focus on session-long learning agents. *In COGNITIVE 2013, The Fifth International Conference on Advanced Cognitive Technologies and Applications*, pages 71–76.

[Math, 2016] Math, C. (2016). The apache commons mathematics library. `http://commons.apache.org/proper/commons-math/`.

[Mendelson, 1990] Mendelson, B. (1990). *Introduction to topology*. Courier Corporation.

[Metropolis et Ulam, 1949] Metropolis, N. et Ulam, S. (1949). The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341. PMID : 18139350.

[Morii et Ito, 2014] Morii, S. et Ito, T. (2014). Agentmr : Concession strategy based on heuristic for automated negotiating agents. *In* Marsa-Maestre, I., Lopez-Carmona, M. A., Ito, T., Zhang, M., Bai, Q. et Fujita, K., éditeurs : *Novel Insights in Agent-based Complex Automated Negotiation*, pages 181–185. Springer Japan, Tokyo.

[Mudgal et Vassileva, 2000] Mudgal, C. et Vassileva, J. (2000). Bilateral negotiation with incomplete and uncertain information : A decision-theoretic approach using a model of the opponent. *In* Klusch, M. et Kerschberg, L., éditeurs : *Cooperative Information Agents IV - The Future of Information Agents in Cyberspace : 4th International Workshop, CIA 2000, Boston, MA, USA, July 7-9, 2000. Proceedings*, pages 107–118, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Myerson, 1979] Myerson, R. B. (1979). Incentive compatibility and the bargaining problem. *Econometrica*, 47(1):61–73.

[Myerson, 1991] Myerson, R. B. (1991). Analysis of incentive in bargaining and mediation. *In* Raiffa, H., éditeur : *Negotiation Analysis*, pages 67–85. University of Michigan Press.

[Myerson et Satterthwaite, 1983] Myerson, R. B. et Satterthwaite, M. A. (1983). Efficient mechanisms for bilateral trading. *Journal of Economic Theory*, 29(2):265 – 281.

[Méhat et Cazenave, 2008] Méhat, J. et Cazenave, T. (2008). Monte Carlo tree search for general game playing. Rapport technique, Université Paris 8.

[Nakatsuka *et al.*, 2000] Nakatsuka, K., Yamaki, H. et Ishida, T. (2000). Market-based network resource allocation with non-tâtonnement process. *In* Zhang, C. et Soo, V.-W., éditeurs : *Design and Applications of Intelligent Agents*, volume 1881 de *Lecture Notes in Computer Science*, pages 143–155. Springer Berlin Heidelberg.

[Nash, 1950] Nash, J. F. (1950). The bargaining problem. *Econometrica : Journal of the Econometric Society*, 2(18):155–162.

[Ndiema *et al.*, 2010] Ndiema, E., Dillian, C. D. et Braun, D. R. (2010). *Interaction and Exchange Across the Transition to Pastoralism, Lake Turkana, Kenya*, chapitre Trade and Exchange : Archaeological Studies from History and Prehistory, pages 95–110. Springer New York, New York, NY.

[Negishi, 1961] Negishi, T. (1961). On the formation of prices. *International Economic Review*, 2(1):122–126.

[Negishi, 1962] Negishi, T. (1962). The stability of a competitive economy : A survey article. *Econometrica*, 30(4):635–669.

[Nickisch et Rasmussen, 2008] Nickisch, H. et Rasmussen, C. E. (2008). Approximations for binary gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078.

[Niemann et Lang, 2009] Niemann, C. et Lang, F. (2009). Assess your opponent : A bayesian process for preference observation in multi-attribute negotiations. *In* Ito, T., Zhang, M., Robu, V., Fatima, S. et Matsuo, T., éditeurs : *Advances in Agent-Based Complex Automated Negotiations*, pages 119–137. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Niimi et Ito, 2016] Niimi, M. et Ito, T. (2016). Agentm. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 235–240. Springer International Publishing.

[Nisan, 2000] Nisan, N. (2000). Bidding and allocation in combinatorial auctions. *In Proceedings of the 2nd ACM conference on Electronic commerce*, EC '00, pages 1–12. ACM.

[Noy *et al.*, 2000] Noy, N. F., Fergerson, R. W. et Musen, M. A. (2000). The knowledge model of protégé-2000 : Combining interoperability and flexibility. *In* Dieng, R. et Corby, O., éditeurs : *Knowledge Engineering and Knowledge Management Methods, Models, and Tools*, volume 1937. Springer Berlin Heidelberg.

[Oprea, 2003] Oprea, M. (2003). The use of adaptive negotiation by a shopping agent in agent-mediated electronic commerce. *In* Mařík, V., Pěchouček, M. et Müller, J., éditeurs : *Multi-Agent Systems and Applications III : 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003 Prague, Czech Republic, June 16–18, 2003 Proceedings*, pages 594–605, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Osborne et Rubinstein, 1990] Osborne, M. J. et Rubinstein, A. (1990). *Bargaining and Markets*. Academic Press.

[Osborne et Rubinstein, 1994] Osborne, M. J. et Rubinstein, A. (1994). *A course in game theory*, chapitre Extensive games with perfect information, pages 89–115. MIT press, 12 édition.

[Oshrat *et al.*, 2009] Oshrat, Y., Lin, R. et Kraus, S. (2009). Facing the challenge of human-agent negotiations via effective general opponent modeling. *In Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '09, pages 377–384, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Parker et Rardin, 2014] Parker, G. R. et Rardin, R. L. (2014). *Discrete optimization*. Elsevier.

[Parkes et Ungar, 2000] Parkes, D. C. et Ungar, L. H. (2000). Iterative combinatorial auctions : Theory and practice. *In* Kautz, H. et Porter, B., éditeurs : *Proceedings of the 17th National Conference on Artificial Intelligence*, pages 74–81. The AAAI Press.

[Ponsen *et al.*, 2010] Ponsen, M. J., Gerritsen, G. et Chaslot, G. (2010). Integrating opponent models with monte-carlo tree search in poker. *In Interactive Decision Theory and Game Theory*, Papers from the AAAI Workshop, pages 37–42. AAAI Press.

[Rasmussen et Williams, 2006a] Rasmussen, C. E. et Williams, C. K. I. (2006a). *Gaussian processes for machine learning*. MIT Press.

[Rasmussen et Williams, 2006b] Rasmussen, C. E. et Williams, C. K. I. (2006b). *Gaussian processes for machine learning*, chapitre Classification, pages 33–77. MIT Press.

[Rassenti *et al.*, 1982] Rassenti, S. J., Smith, V. L. et Bulfin, R. L. (1982). A combinatorial auction mechanism for airport time slot allocation. *The Bell Journal of Economics*, 13(2):402–417.

[Rau *et al.*, 2006] Rau, H., Tsai, M.-H., Chen, C.-W. et Shiang, W.-J. (2006). Learning-based automated negotiation between shipper and forwarder. *Computers & Industrial Engineering*, 51(3): 464 – 481. Special Issue on Selected Papers from the 34th. International Conference on Computers and Industrial Engineering (ICC&IE).

[REL, 2016] REL (2016). 2016 europe working capital survey. Rapport technique, REL.

[Ren et Zhang, 2007] Ren, F. et Zhang, M. (2007). Predicting partners' behaviors in negotiation by using regression analysis. *In* Zhang, Z. et Siekmann, J., éditeurs : *Knowledge Science, Engineering and Management : Second International Conference, KSEM 2007, Melbourne, Australia, November 28-30, 2007. Proceedings*, pages 165–176, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Robinson, 1985] Robinson, M. S. (1985). Collusion and the choice of auction. *The RAND Journal of Economics*, 16(1):141–145.

[Robu et La Poutré, 2005] Robu, V. et La Poutré, H. (2005). Retrieving the structure of utility graphs used in multi-item negotiation through collaborative filtering. *In* ACM, éditeur : *roceedings of the fourth international joint conference on autonomous agents and multiagent systems, AAMAS'05*, pages 280—-287.

[Rodríguez *et al.*, 1997] Rodríguez, J. A., Noriega, P., Sierra, C. et Padget, J. (1997). Fm96.5 a java-based electronic auction house. *In 2nd International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM) '97*, pages 207–224.

[Rolet *et al.*, 2009a] Rolet, P., Sebag, M. et Teytaud, O. (2009a). Boosting active learning to optimality : A tractable monte-carlo, billiard-based algorithm. *In* Buntine, W., Grobelnik, M., Mladenić, D. et Shawe-Taylor, J., éditeurs : *Machine Learning and Knowledge Discovery in Databases : European Conference*, pages 302–317. Springer Berlin Heidelberg.

[Rolet *et al.*, 2009b] Rolet, P., Sebag, M. et Teytaud, O. (2009b). Optimal robust expensive optimization is tractable. *In Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pages 1951–1956. ACM.

[Rubinstein, 1982] Rubinstein, A. (1982). Perfect equilibrium in a bargaining model. *Econometrica*, 50(1):97–109.

[Saaty, 2013] Saaty, T. L. (2013). Analytic hierarchy process. *In* Gass, S. I. et Fu, M. C., éditeurs : *Encyclopedia of Operations Research and Management Science*, pages 52–64. Springer US, Boston, MA.

[Sabater et Sierra, 2002] Sabater, J. et Sierra, C. (2002). Reputation and social network analysis in multi-agent systems. *In Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems : Part 1*, AAMAS '02, pages 475–482, New York, NY, USA. ACM.

[Saha *et al.*, 2005] Saha, S., Biswas, A. et Sen, S. (2005). Modeling opponent decision in repeated one-shot negotiations. *In Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '05, pages 397–403, New York, NY, USA. ACM.

[Sambracos et Yannelis, 1993] Sambracos, E. et Yannelis, D. (1993). On non-tatonnement processes. Mpra paper, University Library of Munich, Germany.

[Samuelson, 1947] Samuelson, P. A. (1947). *Foundations of Economic Analysis*. Harvard University Press.

[Sandholm, 1993] Sandholm, T. (1993). An implementation of the contract net protocol based on marginal cost calculations. *In Proceedings of the Eleventh National Conference on Artificial Intelligence*, AAAI'93, pages 256–262. AAAI Press.

[Sandholm, 2002] Sandholm, T. (2002). Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135(1–2):1–54.

[Sandholm et Lesser, 2001] Sandholm, T. et Lesser, V. R. (2001). Leveled commitment contracts and strategic breach. *Games and Economic Behavior*, 35(1-2):212–270.

[Sandholm, 1999] Sandholm, T. W. (1999). Distributed rational decision making. *In Multiagent Systems*, pages 201–258. MIT Press Cambridge.

[Sandholm et Huai, 2000] Sandholm, T. W. et Huai, Q. (2000). Nomad : Mobile agent system for an internet-based auction house. *IEEE Internet Computing*, 4(2):80o–86.

[Sato et Ito, 2016] Sato, M. et Ito, T. (2016). Whaleagent : Hardheaded strategy and conceder strategy based on the heuristics. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 272–278. Springer International Publishing.

[Schinkel, 2001] Schinkel, M. P. (2001). *Disequilibrium theory : reflections towards a revival of learning*. Thèse de doctorat, Universitaire Pers Maastricht (UPM).

[Schlieper, 1974] Schlieper, U. (1974). Tatonnement and non tatonnement processes of the price mechanism. *Zeitschrift für Nationalökonomie*, 34(1-2):107–124.

[Selman et Gomes, 2006] Selman, B. et Gomes, C. P. (2006). *Encyclopedia of Cognitive Science*, chapitre Hill-climbing Search. John Wiley & Sons, Ltd.

[Şerban *et al.*, 2012] Şerban, L. D., Silaghi, G. C. et Litan, C. M. (2012). Agentfsega : Time constrained reasoning model for bilateral multi-issue negotiations. *In* Ito, T., Zhang, M., Robu, V., Fatima, S. et Matsuo, T., éditeurs : *New Trends in Agent-Based Complex Automated Negotiations*, pages 159–165. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Shardanand et Maes, 1995] Shardanand, U. et Maes, P. (1995). Social information filtering : Algorithms for automating "word of mouth". *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 210–217, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

[Sharp, 1978] Sharp, R. (1978). *The Game of Diplomacy*. A. Barker.

[Shoham et Leyton-Brown, 2008a] Shoham, Y. et Leyton-Brown, K. (2008a). *Multiagent systems : Algorithmic, game-theoretic, and logical foundations*, chapitre Protocols for Strategic Agents : Mechanism Design, pages 273–327. Cambridge University Press.

[Shoham et Leyton-Brown, 2008b] Shoham, Y. et Leyton-Brown, K. (2008b). *Multiagent systems : Algorithmic, game-theoretic, and logical foundations*, chapitre Games with Sequential Actions : Reasoning and Computing with the Extensive Form, pages 117–145. Cambridge University Press.

[Shojaiemehr et Kuchaki Rafsanjani, 2013] Shojaiemehr, B. et Kuchaki Rafsanjani, M. (2013). A multi-agent based model for collective purchasing in electronic commerce. *International Journal of Machine Learning and Cybernetics*, 4(4):409–417.

[Silver *et al.*, 2016] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. et Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

[Sim *et al.*, 2009] Sim, K. M., Guo, Y. et Shi, B. (2009). Blgan : Bayesian learning and genetic algorithm for supporting negotiation with incomplete information. *In IEEE Transactions in Systems, Man and Cybernetics*, volume 39. IEEE Press.

[Simon *et al.*, 2003] Simon, L. K., Thoyer, S., Morardet, S., Goodhue, R. E., Rio, P. et Rausser, G. C. (2003). Structure and bargaining power in multilateral negotiations : Application to water management policies in france. *In 2003 Conference (47th), February 12-14, 2003, Fremantle, Australia*. Australian Agricultural and Resource Economics Society.

[Smith, 1980] Smith, R. G. (1980). The contract net protocol : High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113.

[Sonnenschein, 1972] Sonnenschein, H. (1972). Market excess demand functions. *Econometrica*, 40(3):549–563.

[Ströbel, 2000] Ströbel, M. (2000). A framework for electronic negotiations based on adjusted-winner mediation. *In Proceedings 11th International Workshop on Database and Expert Systems Applications*, pages 1020–1028.

[Sturtevant et Korf, 2000] Sturtevant, N. R. et Korf, R. E. (2000). On pruning techniques for multi-player games. *AAAI/IAAI*, 49:201–207.

[Sylvain *et al.*, 2014] Sylvain, A., Doniec, A., Mandiau, R. et Lecoeuche, S. (2014). Purchase intention based model for a behavioural simulation of sale space. *In 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 3, pages 318–324.

[Szöllősi-Nagy *et al.*, 2016] Szöllősi-Nagy, B., Festen, D. et Skarżyńska, M. M. (2016). A greedy coordinate descent algorithm for high-dimensional nonlinear negotiation. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, volume 638 de *Studies in Computational Intelligence*, pages 249–260. Springer International Publishing.

[Tedeschi *et al.*, 2012] Tedeschi, G., Gallegati, M., Mignot, S. et Vignes, A. (2012). Lost in transactions : The case of the boulogne s/mer fish market. *Physica A : Statistical Mechanics and its Applications*, 391(4):1400 – 1407.

[Tieben, 2009] Tieben, B. (2009). *The Concept of Equilibrium in Different Economic Traditions : A Historical Investigation*. Rozenberg Publishers.

[Tsvetovatyy *et al.*, 1997] Tsvetovatyy, M., Gini, M., Mobasher, B., Ski, Z. W. et Ski, W. (1997). Magma : An agent based virtual market for electronic commerce. *Applied Artificial Intelligence*, 11(6):501–523.

[Turban *et al.*, 2015] Turban, E., King, D., Lee, J., Viehland, D. et Trurban, D. C. (2015). *Electronic commerce : A managerial perspective*. Springer Texts in Business and Economics. Springer International Publishing, 8th édition.

[van Galen Last, 2012] van Galen Last, N. (2012). Agent smith : Opponent model estimation in bilateral multi-issue negotiation. *In* Ito, T., Zhang, M., Robu, V., Fatima, S. et Matsuo, T., éditeurs : *New Trends in Agent-Based Complex Automated Negotiations*, pages 167–174. Springer Berlin Heidelberg, Berlin, Heidelberg.

[van Krimpen *et al.*, 2013] van Krimpen, T., Looije, D. et Hajizadeh, S. (2013). Hardheaded. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, pages 223–227. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Varian et Harris, 2013] Varian, H. R. et Harris, C. (2013). Vcg in theory and practice. Rapport technique, Google inc.

[Vickrey, 1961] Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37.

[Walras, 1954] Walras, L. (1954). *Elements of pure economics*. Published for the American Economic Association and the Royal Economic Society by Allen and Unwin.

[Weintraub, 1979] Weintraub, E. R. (1979). *Microfoundations : the compatibility of microeconomics and macroeconomics*. Cambridge University Press.

[Williams et Barber, 1998] Williams, C. K. I. et Barber, D. (1998). Bayesian classification with gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12): 1342–1351.

[Williams *et al.*, 2011] Williams, C. R., Robu, V., Gerding, E. H. et Jennings, N. R. (2011). Using gaussian processes to optimise concession in complex negotiations against unknown opponents. *In Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume One*, IJCAI'11, pages 432–438. AAAI Press.

[Williams *et al.*, 2012] Williams, C. R., Robu, V., Gerding, E. H. et Jennings, N. R. (2012). Iamhaggler : A negotiation agent for complex environments. *In* Ito, T., Zhang, M., Robu, V., Fatima, S. et Matsuo, T., éditeurs : *New Trends in Agent-Based Complex Automated Negotiations*, pages 151–158. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Williams *et al.*, 2013] Williams, C. R., Robu, V., Gerding, E. H. et Jennings, N. R. (2013). Iamhaggler2011 : A gaussian process regression based negotiation agent. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, pages 209–212. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Wilson, 2001] Wilson, C. A. (2001). Mediation and the nash bargaining solution. *Review of Economic Design*, 6(3):353–370.

[Winoto, 2007] Winoto, P. (2007). *Modified bargaining protocols for automated negotiation in open multi-agent systems*. Thèse de doctorat, University of Saskatchewan Saskatoon.

[Wurman *et al.*, 1998] Wurman, P. R., Wellman, M. P. et Walsh, W. E. (1998). The michigan internet auctionbot : A configurable auction server for human and software agents. *In Proceedings of the Second International Conference on Autonomous Agents*, AGENTS '98, pages 301–308. ACM.

[Yao, 1982] Yao, A. C. (1982). Protocols for secure computations. *In 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 160–164.

[Yao, 1986] Yao, A. C. (1986). How to generate and exchange secrets. *In 27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 162–167.

[Yu et Singh, 2002] Yu, B. et Singh, M. P. (2002). An evidential model of distributed reputation management. *In Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems : Part 1*, AAMAS '02, pages 294–301, New York, NY, USA. ACM.

[Yu *et al.*, 2013] Yu, C., Ren, F. et Zhang, M. (2013). An adaptive bilateral negotiation model based on bayesian learning. *In* Ito, T., Zhang, M., Robu, V. et Matsuo, T., éditeurs : *Complex Automated Negotiations : Theories, Models, and Software Competitions*, pages 75–93. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Zafari et Nassiri-Mofakham, 2016] Zafari, F. et Nassiri-Mofakham, F. (2016). Bravecat : Iterative deepening distance-based opponent modeling and hybrid bidding in nonlinear ultra large bilateral multi issue negotiation domains. *In* Fukuta, N., Ito, T., Zhang, M., Fujita, K. et Robu, V., éditeurs : *Recent Advances in Agent-based Complex Automated Negotiation*, pages 285–293. Springer International Publishing, Cham.

[Zeng et Sycara, 1998] Zeng, D. et Sycara, K. (1998). Bayesian learning in negotiation. *International Journal of Human-Computer Studies*, 48(1):125–141.

[Zhang et Liu, 2016] Zhang, L. et Liu, Q. (2016). An automated multi-issue negotiation mechanism based on intelligent agents in e-commerce. *Journal of Advanced Management Science*, 4(2):172–175.

[Ziegler et Lausen, 2004] Ziegler, C.-N. et Lausen, G. (2004). Spreading activation models for trust propagation. *In e-Technology, e-Commerce and e-Service, 2004. EEE '04. 2004 IEEE International Conference on*, pages 83–97.

[Zitzler *et al.*, 2001] Zitzler, E., Laumanns, M. et Thiele, L. (2001). Spea2 : Improving the strength pareto evolutionary algorithm. Rapport technique 103, Eidgenössische Technische Hochschule Zürich.