Du raisonnement social chez les agents : une approche fondée sur la théorie de la dépendance

Jaime Simao Sichman

▶ **To cite this version:**

Jaime Simao Sichman. Du raisonnement social chez les agents : une approche fondée sur la théorie de la dépendance. Interface homme-machine [cs.HC]. Institut National Polytechnique de Grenoble - INPG, 1995. Français. NNT : . tel-00005063

HAL Id: tel-00005063

https://theses.hal.science/tel-00005063

Submitted on 24 Feb 2004

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

présentée par

Jaime Simão Sichman
MsC Engenharia Elétrica, Escola Politécnica da USP

pour obtenir le grade de Docteur
de l'Institut National Polytechnique de Grenoble
(Arrêté ministériel du 30 mars 1992)

Spécialité : INFORMATIQUE

Du Raisonnement Social Chez les Agents

Une Approche Fondée sur la Théorie de la Dépendance

Soutenue le 05/09/95

Composition du jury :

Président :	M. Augustin Lux
Rapporteurs :	M. Cristiano Castelfranchi
	M. Helder Coelho
Examinateurs :	M. Jacques Ferber
	M. Jean Sallantin
	M. Yves Demazeau

Thèse préparée au sein du **L**aboratoire d'**I**nformatique **F**ondamentale et d'**I**ntelligence **A**rtificielle

Résumé

Cette thèse présente le modèle d'un mécanisme de raisonnement social fondé sur la théorie de la dépendance. Ce modèle permet à un agent de raisonner sur autrui et plus particulièrement de calculer ses relations et situations de dépendance. Un agent est dépendant d'un autre si celui-ci peut l'aider/l'empêcher d'atteindre un de ses buts. Nous considérons notre mécanisme de raisonnement social comme un composant essentiel pour la conception d'agents artificiels réellement autonomes, évoluant dans un univers multi-agents ouvert. La notion d'ouverture désigne la capacité d'ajouter ou de retirer dynamiquement dans le système des agents. Comme dans ces systèmes l'organisation des agents ne peut pas être spécifiée pendant la phase de conception, la résolution coopérative de problèmes est fondée sur la formation dynamique de coalitions. Dans ce contexte, des agents doivent être capables de s'adapter aux changements dynamiques du système, en particulier en évaluant pendant la phase de résolution si leurs buts sont réalisables et si leurs plans sont exécutables. Comme nous ne supposons pas que les agents soient bienveillants, notre modèle fournit un critère pour évaluer les partenaires les plus susceptibles d'accepter une proposition de coalition. Enfin, comme dans ces systèmes des agents n'ont pas généralement une représentation complète et correcte les uns des autres, notre modèle leur permet de détecter une inconsistance au niveau de la société et de choisir un contexte à être maintenue. Nous avons implémenté ce mécanisme de raisonnement social en utilisant une programmation orientée objet. Nous l'avons utilisé pour développer deux applications, le simulateur DEPNET et le système DEPINT, qui illustrent son utilisation selon deux perspectives scientifiques différentes. D'une part, selon une perspective de simulation sociale, notre modèle fournit un outil informatique permettant l'analyse et la prédiction des divers schémas intéressants d'interaction sociale, et l'évaluation du pouvoir social des agents. D'autre part, selon une perspective de résolution de problèmes, notre modèle peut être utilisé pour concevoir dynamiquement l'organisation des agents dans un contexte de systèmes multi-agents ouverts.

Mots Clés : raisonnement social, raisonnement sur autrui, intelligence artificielle distribuée, systèmes multi-agents, intégration de systèmes, systèmes ouverts.

Abstract

This thesis presents the model of a social reasoning mechanism based on dependence theory. This model enables an agent to reason about the others, in particular to calculate his dependence relations and dependence situations. An agent is said to be dependent on another if the latter can help/prevent him to achieve one of his goals. We consider our social reasoning mechanism as an essential building block for the design of really autonomous artificial agents, which are immersed in an open multi-agent world. By open, we mean that agents may enter or leave the agency at any moment. In such systems, as the organisation of the agents can not be conceived at design time, the cooperative problem solving paradigm is based on dynamic coalition formation. In this context, agents must be able to adapt themselves to dynamically changing conditions, by evaluating at execution time if their goals are achievable and if their plans are feasible. As we do not suppose that agents are benevolent, our model proposes a criterion to evaluate which partners are more susceptible to accept a proposition of coalition. Finally, as in these kind of systems agents usually do not have a complete and correct representation of each other, our model helps them to detect an agency level inconsistency and to choose a context to be maintained. We have implemented our social reasoning mechanism using an object-oriented approach, and we have used it to develop two applications, the DEPNET simulator and the DEPINT system, which illustrate respectively its usage in two different scientific perspectives. On one hand, concerning social simulation, our model provides a computational tool for the analysis and prediction of the occurrence of several interesting patterns of social interactions, and for the evaluation of the agents' social power. On the other hand, with respect to problem solving, our model can be used to design dynamic agents' organizations in a context of open multi-agent systems.

Keywords : social reasoning, reasoning about the others, distributed artificial intelligence, multi-agent systems, system integration, open systems.

Resumo

Esta tese propõe um modelo de um mecanismo de raciocínio social baseado na teoria da dependência. Tal modelo permite a um agente raciocinar sobre os outros, em particular calcular suas relações e situações de dependência. Um agente depende de um outro se este último pode lhe facilitar/prejudicar a obtenção de um de seus objetivos. Nós consideramos nosso mecanismo de raciocínio social como um elemento fundamental para a concepção de agentes artificiais realmente autônomos, que estejam imersos num universo multi-agentes aberto. A noção de abertura denota o fato de que os agentes podem entrar ou sair do sistema em qualquer momento. Em tais sistemas, como a organização dos agentes não pode ser especificada durante a sua fase de concepção, o modelo cooperativo de resolução de problemas se baseia na formação dinâmica de coligações. Neste contexto, os agentes devem ser capazes de se adaptar a mudanças dinâmicas, avaliando durante a fase de execução se seus objetivos são realizáveis e se seus planos são executáveis. Como nós não supomos que os agentes sejam benevolentes, nosso modelo propõe um critério para avaliar os agentes mais susceptíveis a aceitar uma proposta de coligação. Finalmente, como em tais sistemas os agentes normalmente não têm uma representação completa e correta uns dos outros, nosso modelo lhes permite detectar uma inconsistência global e escolher un contexto a ser mantido. Nós implementamos este mecanismo de raciocínio social utilizando uma programação orientada a objetos, e o utilisamos para desenvolver duas aplicações, o simulador DEPNET e o sistema DEPINT, que ilustram o seu interesse sob duas perspectivas científicas diversas. Por um lado, considerando uma perspectiva de simulação social, tal modelo fornece uma ferramenta computacional para a análise e predição de diversos tipos de interações sociais interessantes e para a avaliação do poder social dos agentes. Por outro lado, no que diz respeito à resolução de problemas, nosso modelo pode ser utilizado para a concepção dinâmica de organizações de agentes num contexto de sistemas multi-agentes abertos.

Palavras Chave : raciocínio social, raciocínio sobre os outros, inteligência artificial distribuída, sistemas multi-agentes, integração de sistemas, sistemas abertos.

Riassunto

Questa tesi presenta un modello di meccanismo di raggionamento sociale basato sulla teoria della dipendenza. Tale modello abilita un agente a raggionare su altri agenti, in particolare ad estimare le sue relazioni e situazioni di dipendenza. Un agente é detto dipendente da un altro agente se quest'último può aiutare/impedire il raggiungimento dei suoi scòpi. Riteniamo che il nostro meccanismo di raggionamento sociale sia un essenziale contributo alla teoria degli agenti artificiali autonomi in un mondo multi-agenti aperto. Il concetto de l'apertura significa che gli agenti possono entrare e/o uscire dal sistema in qualsiasi istante. In tali sistemi, come l'organizzazione degli agenti non può essere specificata durante la fasi di proggetazione, il modello cooperativo di risoluzione dei problemi si basa nella formazione dinamica di coalizione. In questo contexto, gli agenti debbono essere capabili di adattarsi a modifique dinamique, valutando durante la fasi di esecuzione si i suoi scòpi sono raggiungibili e si i suoi piani sono eseguibili. Come noi non supponiamo che gli agenti siano benevolenti, il nostro modello propone un criterio di valutazione degli agenti piu suscettibili ad accettare una proposta di coalizione. Finalmente, come in questi sistemi gli agenti non hanno una rappresentazione completa e corretta nei confronti degli altri agenti, questo modello permette un agente de scoprire una inconsistenza globale e di scegliere un contexto ad essere seguito. Noi abbiamo applicato il modello di programmazione orientata ad oggetti per implementare questo meccanismo de raggionamento sociale e l'abbiamo utilizzato per sviluppare le due applicazione, il simulatore DEPNET ed il sistema DEPINT, che illustrano sua importanza su due prospettive scientifique diverse. Da un canto, considerando una prospettiva di simulazione sociale, il modello descritto permette uno studio approfondito dei diversi tipi di interazioni sociali e la valutazione del potere sociale degli agenti. D'altro canto, il modello da noi proposto dá la possibilitá di organizzare dinamicamente gli agenti nel contexto dei sistemi multi-agenti aperti.

Parole Chiave : raggionamento sociale, raggionamento su gli altri, intelligenza artificiale distribuita, sistemi multi-agenti, integrazione dei sistemi, sistemi aperti.

Remerciements

Je tiens à remercier :

Monsieur Augustin Lux, Professeur à l'Institut National Polytechnique de Grenoble, pour m'avoir fait l'honneur de bien vouloir présider ce jury. Monsieur Lux a aussi supervisé mes travaux depuis mon arrivée au LIFIA/IMAG, et son aide a été inestimable pendant mes premiers mois en France. Je tiens à lui exprimer toute ma gratitude ;

Messieurs Cristiano Castelfranchi, Directeur de Recherche CNR à l'Istituto di Psicologia, Rome, Italie et Helder Coelho, Professeur à l'Universidade Técnica de Lisboa, Portugal, pour avoir accepté de juger ce travail, pour le temps qu'ils y ont consacré et pour les remarques constructives qu'ils ont bien voulu porter sur mes recherches ;

Messieurs Jacques Ferber, Professeur à l'Université Paris VI, et Jean Sallantin, Directeur de Recherche CNRS au LIRMM, pour avoir accepté de faire partie du jury de cette thèse ;

Monsieur Yves Demazeau, Chargé de Recherche CNRS au LIFIA/IMAG, qui a encadré cette thèse. Je le remercie pour m'avoir accueilli dans le groupe MAGMA, et pour la confiance qu'il m'a accordée. S'il existe une personne à qui je dois mes connaissances dans le domaine des systèmes multi-agents, c'est bien lui. Je lui suis reconnaissant aussi pour m'avoir permis d'élargir mes contacts scientifiques dans le domaine, soit en rendant possible ma participation dans plusieurs colloques et conférences françaises, européennes et internationales, soit en étant d'accord dès le début pour le développement de cette thèse dans un cadre européen.

Cette thèse ainsi a été le fruit d'une coopération scientifique établie entre le LIFIA/IMAG, à Grenoble et l'Istituto di Psicologia del CNR, à Rome. Particulièrement, je voudrais exprimer toute ma gratitude à Monsieur Castelfranchi qui m'a donné la possibilité de séjourner pendant quelques mois en 1993 et 1994 à Rome, dans le cadre de cette coopération scientifique. Je voudrais aussi remercier tous les membres du groupe PSCS (Amedeo Cesta, Daniela D'Aloisi, Maria Micelli, Paola Rizzo and Vittorio Giannini) qui m'ont beaucoup stimulé durant mon séjour à Rome. Un remerciement tout particulier est adressé à Rosaria Conte, pour sa gentillesse, ses conseils, sa rigueur scientifique, son esprit d'analyse et pour son aide en de nombreuses occasions, professionnellement ou personnellement. J'ai eu la chance de pouvoir partager ses qualités humaines pendant ce séjour, et je tiens à lui exprimer toute mon amitié.

Je remercie le Departamento de Engenharia de Computação e Sistemas Digitais (PCS) de l'Escola Politécnica da Universidade de São Paulo (USP) pour m'avoir accordé un détachement de mes fonctions d'enseignant de 1991 à 1995. La recherche décrite dans ce manuscrit a été financée par la Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Brésil, sous la bourse numéro 91/1943-5, à laquelle j'exprime aussi ma gratitude.

Cette thèse aborde quelques aspects à la fois formels, algorithmiques et applicatifs. De cette façon, j'ai eu la chance de bénéficier des conseils de personnes certainement plus compétentes que moi dans ces domaines. Tout d'abord, en ce qui concerne les aspects formels, je voudrais remercier Messieurs Michael Wooldridge (University of Manchester, UK), Ricardo Caferra, Philipe Schnoebelen et Stephan Demri (LIFIA/IMAG) pour leurs conseils scientifiques. En ce qui concerne les aspects d'implémentation et d'algorithmique, je remercie fortement Monsieur Eleri Cardozo, Professeur à la Faculdade de Engenharia da Universidade de Campinas, Brésil, dont le support technique a été inestimable lors de l'installation à Grenoble du système DPSK+P, sur lequel la plate-forme multi-agents du groupe MAGMA est actuellement installée. Je tiens à remercier aussi tous les étudiants (Philippe Populaire, Janick Blanc et Guillaume Giles) qui ont implémentés des améliorations et extensions de cette plate-forme.

Je tiens aussi à remercier :

Monsieur Philippe Jorrand, Directeur du LIFIA/IMAG, qui a accepté de m'accueillir dans ce laboratoire ;

Les membres du LIFIA/IMAG, pour l'ambiance chaleureuse dont j'ai pu bénéficier pendant toutes ces années. Je voudrais remercier particulièrement Monsieur Henri Saya, ingénieur systèmes du laboratoire, pour avoir réglé les problèmes que j'ai pu avoir pendant l'implémentation de mes travaux ;

Les membres de l'équipe MAGMA (Jean-Luc Koning, Michel Occelo, Luis Alvares, Uma Garimella, Christof Baeijs, Nils Ferrand), ainsi que Marie-Hélène Stefanini, qui ont accepté gentiment de lire ce manuscrit et de corriger "*tous*" les fautes de français que j'ai pu produire, et qui m'ont aussi soutenu dans les moments difficiles que tout travail de ce genre comporte ;

Monsieur James L. Crowley, Professeur à l'Institut National Polytechnique de Grenoble, qui m'a accepté dans l'équipe PRIMA, lorsque le groupe MAGMA n'existait pas encore. Je remercie aussi tous les membres de cette équipe, en particulier Olivier Causse, Patrick Regnier, Christophe Discours, Matthew Turk et Philippe Bobet, qui m'ont beaucoup aidé à "vivre à la française" lors de mon arrivée ;

Les membres du groupe PLEIAD et du groupe MARCIA du PRC-GDR IA, ainsi que les membres de la communauté internationale du domaine des systèmes multi-agents, dont les discussions scientifiques ont constitué un élément important de réflexion dans mes recherches et avec lesquels j'ai eu le plaisir de partager quelques moments pendant ces années ;

Les membres de la communauté brésilienne à Grenoble, que je ne peux pas tous énumérer ici, qui m'ont beaucoup aidé à relativiser la distance entre la France et mon pays: "obrigado" ;

Monsieur Wilfried Pfister, qui a développé un style LaTeX 2ε pour les thèses au LIFIA, avec lequel ce manuscrit a été produit. J'ai eu la chance de pouvoir compter sur son support technique, parfois de façon abusive, et je lui suis très reconnaissant de sa gentillesse.

Valérie et Jean-Claude (merci à la factrice !) ont su me démontrer que même en France on peut trouver des voisins très aimables. Je les remercie aussi pour avoir jeté un dernier coup d'œil sur mes fautes de français.

Olivier Boissier sait combien je le suis redevable pour cette thèse. Je ne pourrais pas décrire ici l'apport qu'il m'a donné, dans les situations les plus diverses, d'un point de vue théorique, d'implémentation et plutôt humain. Qu'il sache que je le remercie de tout mon cœur, et qu'il a une place dans l'ensemble de mes amis les plus chers.

J'adresse un remerciement tout spécial à mes parents, auxquels je dois presque tout dans ma vie. Ils ont non seulement su supporter mon absence prolongée pendant toutes ces années, mais aussi ils m'ont donné le support moral, en m'encourageant pendant des moments difficiles. Je tiens à leur exprimer tout mon amour et mon respect.

Pendant le temps d'une thèse, il est certain que beaucoup de choses se passent dans la vie d'une personne et du monde. Par exemple, le monde a connu (heureusement!) son unique tetra-champion mondial de football, mais il a perdu aussi (malheureusement!) son meilleur pilote de Formule 1. Quant à moi, pendant cette période, j'ai eu la chance de rencontrer une personne très spéciale, qui m'a rendu le plus heureux des hommes en acceptant de devenir ma femme.

Isabella peut-être ne sait pas combien je lui suis redevant pour cette thèse. Tout d'abord, c'est à cause d'elle que j'ai eu l'idée de rester quelques mois à Rome. Elle a su me soutenir dans les moments les plus difficiles, en particulier pendant mes absences prolongées de la maison lors des derniers mois de rédaction de ce manuscrit. Enfin, je dois avouer que je ne suis pas certain d'être capable de supporter une mauvaise humeur telle que la mienne lors de la fin de mes travaux. Qu'elle sache que je me sens vraiment heureux de pouvoir partager mes jours à ses cotés et que mon amour et mon admiration pour elle ne font que s'agrandir au fur et à mesure que le temps passe ...

Table des matières

Résumé i

Abstract iii

Resumo v

Riassunto vii

Remerciements ix

1 Introduction 1
- 1.1 Motivations . . . 1
- 1.2 Objectifs scientifiques . . . 2
- 1.3 Principes adoptés . . . 5
- 1.4 Remarques méthodologiques . . . 6
  - 1.4.1 Dualité informatique/science cognitive . . . 6
  - 1.4.2 Orientation modèles/systèmes . . . 8
- 1.5 Démarche suivie . . . 9
- 1.6 Organisation du manuscrit . . . 12

I Systèmes multi-agents 15

2 Intelligence artificielle distribuée 17
- 2.1 Caractérisation du domaine . . . 17
  - 2.1.1 Problématique abordée . . . 18
  - 2.1.2 Bref historique . . . 21
  - 2.1.3 Avantages de l'approche . . . 21
  - 2.1.4 Principaux axes de recherche . . . 22
  - 2.1.5 Organisation et interaction entre agents . . . 22
- 2.2 Résolution distribuée de problèmes et systèmes multi-agents . . . 23
  - 2.2.1 Première comparaison entre les sous-domaines . . . 23
  - 2.2.2 Deuxième comparaison entre les sous-domaines . . . 26

2.2.3 Approche Suivie 28
2.3 Modèles d'agents en univers multi-agents 29
2.3.1 Définition d'un agent 29
2.3.2 Agents cognitifs et agents réactifs 30
2.3.3 Exemples d'applications 31
2.4 Synthèse 32

3 Conception des organisations 33
3.1 Approche statique et approche dynamique 33
3.2 Coalitions entre agents 35
3.3 Importance du raisonnement social 36
3.4 Modèle d'agent 37
3.5 Modèle de société 40
3.6 Synthèse 41

II Mécanisme de raisonnement social 43

4 Théorie de la dépendance 45
4.1 Modèles d'interaction sociale 46
4.1.1 Modèles fondés sur les structures organisationelles 47
4.1.2 Modèles fondés sur l'utilité 48
4.1.3 Modèles fondés sur la complémentarité 50
4.2 Autonomie d'un agent 50
4.3 Interférence, action et interaction sociales 52
4.3.1 Interférence sociale 52
4.3.2 Action sociale 53
4.3.3 Interaction sociale 54
4.4 Adoption de buts 55
4.5 Pouvoir social et relations de dépendance 57
4.5.1 Pouvoir social 57
4.5.2 Relations de dépendance 58
4.6 Types de relations de dépendance 59
4.6.1 Ou-dépendance et et-dépendance 59
4.6.2 Dépendance unilatérale et dépendance bilatérale 60
4.6.3 Dépendance externe et dépendance transitive 60
4.7 Des relations de dépendance aux interactions sociales 61
4.8 Discussion 62

5 Conception du modèle 65
5.1 Description externe 65
5.1.1 Composition 66
5.1.2 Sources d'information 68
5.2 Définitions préliminaires 69

5.2.1 Agents sujet, objet, tiers et source . . . 69
5.2.2 Types d'autonomie et types de dépendance . . . 70
5.3 Langage interne . . . 71
5.3.1 Notions de base . . . 71
5.3.2 Interfaçage avec la description externe . . . 72
5.3.3 Notions auxiliaires . . . 72
5.3.4 Notions sur les plans . . . 73
5.3.5 Notions d'autonomie . . . 74
5.3.6 Relations de dépendance . . . 74
5.3.7 Dépendance mutuelle et dépendance réciproque . . . 75
5.3.8 Quelques simplifications . . . 75
5.3.9 Situations de but . . . 77
5.3.10 Situations de dépendance . . . 78
5.4 Discussion . . . 79

6 Effet sur l'adaptation **81**
6.1 Plans exécutables et buts réalisables . . . 81
6.2 Choix de buts . . . 82
6.2.1 Procédure de choix de buts . . . 82
6.2.2 Critère de choix de buts . . . 84
6.2.3 Gestion de choix de buts par le modèle d'agent . . . 85
6.2.4 A propos de la genèse de buts . . . 87
6.3 Choix de plans . . . 87
6.3.1 Procédure de choix de plans . . . 87
6.3.2 Critère de choix de plans . . . 88
6.3.3 Gestion de choix de plans par le modèle d'agent . . . 90
6.3.4 A propos de la planification . . . 90
6.4 Discussion . . . 91

7 Effet sur la formation de coalitions **93**
7.1 Choix de partenaires . . . 93
7.1.1 Procédure de choix de partenaires . . . 94
7.1.2 Critère de la source de plans . . . 94
7.1.3 Critère de la nature de la dépendance . . . 95
7.1.4 Critère de choix de partenaires . . . 97
7.1.5 Gestion de choix de partenaires par le modèle d'agent . . . 99
7.1.6 A propos du principe de bienveillance . . . 101
7.2 Acceptation de partenaires . . . 101
7.2.1 Procédure d'acceptation de partenaires . . . 102
7.2.2 Critère d'acceptation de partenaires . . . 103
7.2.3 Gestion d'acceptation de partenaires par le modèle d'agent 104
7.3 Evolution des choix de partenaires . . . 104
7.3.1 Scénario à t=0 . . . 105
7.3.2 Scénario à t=1 . . . 106

7.3.3 Scénario à t=2 . . . 107
7.3.4 Scénario à t=3 . . . 108
7.3.5 Scénario à t=4 . . . 110
7.4 Discussion . . . 111

8 Effet sur la révision de croyances 113
8.1 Modélisation des croyances . . . 114
8.1.1 Approches existantes . . . 114
8.1.2 Approche suivie . . . 114
8.1.3 Modèle déductif de croyances . . . 115
8.2 Langage externe . . . 117
8.2.1 Notions de base . . . 117
8.2.2 Hypothèse de compatibilité de description externe . . . 118
8.3 Inconsistance au niveau de la société . . . 118
8.3.1 Analyse préliminaire des résultats couplés . . . 118
8.3.2 Croyances incomplètes et croyances fausses . . . 119
8.3.3 Analyse détaillée des résultats couplés . . . 124
8.4 Révision de la représentation des autres . . . 127
8.4.1 Révision de croyances dans un contexte mono-agent . . . 128
8.4.2 Révision de croyances dans un contexte multi-agents . . . 129
8.5 Choix de contexte . . . 130
8.5.1 Procédure de choix de contexte . . . 130
8.5.2 Critère de choix de contexte . . . 130
8.5.3 Gestion de choix de contexte par le modèle d'agent . . . 132
8.6 Discussion . . . 133

III Implémentation et applications 135

9 Implémentation du modèle 137
9.1 Description externe . . . 138
9.2 Réseaux de dépendance . . . 143
9.3 Situations de but . . . 150
9.4 Situations de dépendance . . . 151

10 Simulateur DEPNET 155
10.1 Description du simulateur . . . 155
10.2 Simulation d'un environnement de recherche . . . 158
10.2.1 Situation initiale du laboratoire d'informatique . . . 158
10.2.2 Situation initiale du laboratoire d'électronique . . . 164
10.2.3 Situation finale du laboratoire d'informatique . . . 167
10.2.4 Situation finale du laboratoire d'électronique . . . 172
10.3 Discussion . . . 172

11 Système DEPINT **173**
11.1 Plate-forme du groupe MAGMA 173
11.1.1 Description générale 174
11.1.2 Interactions entre agents 174
11.1.3 Protocoles d'interaction 175
11.1.4 Intégration orientée agent 176
11.2 Protocoles d'interaction du système DEPINT 177
11.2.1 Protocole de présentation 177
11.2.2 Protocole de sortie 178
11.2.3 Protocole de formation de coalitions 178
11.3 Implémentation du modèle d'agent 180
11.3.1 Quelques simplifications 180
11.3.2 Cycles d'activation des mécanismes internes 181
11.4 Exemples d'utilisation 185
11.4.1 Autonomie 185
11.4.2 Coalition non-réussie 187
11.4.3 Coalition réussie 193
11.4.4 Révision de croyances 197
11.5 Discussion 203

12 Conclusion **205**
12.1 Problèmes abordés 205
12.2 Perspectives 206
12.2.1 Modèle de raisonnement social 206
12.2.2 Critères de choix de buts, plans et partenaires 207
12.2.3 Révision de croyances 208
12.2.4 Implémentation et applications 209
12.3 Conception statique des organisations 209

Annexes **213**

A Preuves **213**
A.1 Remarques sur les preuves 213
A.2 Langage interne 215
A.3 Hypothèse de commutativité du langage externe 216
A.4 Notions de base 216
A.5 Notions auxiliaires 218
A.6 Notions sur les plans 222
A.7 Notions d'autonomie 226
A.8 Relations de dépendance 228
A.9 Dépendance mutuelle et dépendance réciproque 230
A.10 Situations de dépendance 232

B Description de l'organisation du système EB **237**
B.1 Description du système 237
B.2 Application du modèle 240
B.3 Résultats de simulation 242

C Description de l'organisation du système MAVI **249**
C.1 Description du système 249
C.2 Application du modèle 251
C.3 Résultats de simulation 253

D Abréviations **255**

E Bibliographical remarks **257**

Bibliographie **263**

Liste des figures

1.1 Dualité informatique/science cognitive 7
1.2 Extension de la méthodologie MAD [CAM94] 9
1.3 Démarche suivie 10

2.1 Approche résolution distribuée de problèmes 24
2.2 Approche système multi-agents 26

3.1 Formation de coalitions 36
3.2 Modèle d'agent 39
3.3 Modèle de société 41

4.1 Modèles d'interaction sociale 47
4.2 Dilemme du prisonnier [Axe84] 49
4.3 Aspects cognitifs et pré-cognitifs de l'action sociale [CC92] 61

5.1 Description externe 68

6.1 Critère de choix de buts 85
6.2 Gestion de choix de buts par le modèle d'agent 86
6.3 Critère de choix de plans 89
6.4 Gestion de choix de plans par le modèle d'agent 91

7.1 Critère de la source de plans 95
7.2 Critère de la nature de la dépendance 96
7.3 Critère de choix de partenaires 98
7.4 Gestion de choix de partenaires par le modèle d'agent 100
7.5 Premier exemple du monde des blocs 105

8.1 Modèle déductif de croyances [Kon86] 115
8.2 Croyances incomplètes et croyances fausses 122
8.3 Critère de choix de contexte 132
8.4 Gestion de choix de contexte par le modèle d'agent 133

9.1 Deuxième exemple du monde des blocs 137
9.2 Classes composantes de la description externe 139
9.3 Exemple d'instanciation d'une description externe 142

9.4 Implémentation inclusive des réseaux de dépendance 143
9.5 Classes composantes du réseau de dépendance 144
9.6 Exemple d'instanciation d'un réseau de dépendance 149
9.7 Algorithme pour le calcul des situations de but 151
9.8 Algorithme pour le calcul des situations de dépendance 152

10.1 Simulateur DEPNET 157

11.1 Plate-forme du groupe MAGMA 174
11.2 Protocole de présentation 177
11.3 Protocole de sortie 178
11.4 Protocole de formation de coalitions 178
11.5 Comportement actif 182
11.6 Comportement passif 184

B.1 Système EB 238

C.1 Système MAVI 250

Liste des tables

2.1 Comparaison entre les approches RDP et SMA [DR94] 28

8.1 Inconsistance au niveau de la société 119
8.2 Analyse détaillée des résultats couplés 126

Liste des exemples

7.1 Description externe à t=0 105
7.2 Relations de dépendance à t=0 106
7.3 Situations de dépendance à t=0 106
7.4 Description externe à t=1 107
7.5 Situations de dépendance à t=1 107
7.6 Description externe à t=2 107
7.7 Situations de dépendance à t=2 108
7.8 Description externe à t=3 108
7.9 Relations de dépendance à t=3 109
7.10 Situations de dépendance à t=3 109
7.11 Description externe à t=4 110
7.12 Relations de dépendance à t=4 110
7.13 Situations de dépendance à t=4 110

8.1 Premier exemple d'inconsistance 121
8.2 Deuxième exemple d'inconsistance 123
8.3 Troisième exemple d'inconsistance 123

9.1 Description externe de l'exemple du monde des blocs 138
9.2 Exemple de code pour l'insertion d'un agent 141

10.1 Description externe initiale du laboratoire d'informatique 159
10.2 Réseaux de dépendance initiaux du laboratoire d'informatique . . 160
10.3 Situations de but initiales du laboratoire d'informatique 162
10.4 Situations de dépendance initiales du laboratoire d'informatique . 162
10.5 Description externe du laboratoire d'électronique 164
10.6 Réseaux de dépendance initiaux du laboratoire d'électronique . . 165
10.7 Situations de but initiales du laboratoire d'électronique 167
10.8 Situations de dépendance initiales du laboratoire d'électronique . 167
10.9 Réseaux de dépendance finaux du laboratoire d'informatique . . 167
10.10 Situations de dépendance finales du laboratoire d'informatique . 170

11.1 Exemple d'autonomie 185
11.2 Premier exemple de proposition de coalition 187
11.3 Exemple de refus 189

11.4 Exemple de traitement de refus 191
11.5 Deuxième exemple de proposition de coalition 193
11.6 Exemple d'acceptation 196
11.7 Exemple de traitement d'acceptation 197
11.8 Exemple d'inférence 197
11.9 Troisième exemple de proposition de coalition 199
11.10 Exemple de détection d'inconsistance 201
11.11 Exemple de révision de croyances 202

B.1 Description externe du système EB 240
B.2 Réseaux de a-dépendance du système EB 243
B.3 Réseaux de r-dépendance du système EB 244
B.4 Situations de dépendance du système EB 245

C.1 Description externe du système MAVI 251
C.2 Réseaux de a-dépendance du système MAVI 253
C.3 Situations de dépendance du système MAVI 254

Chapitre 1

Introduction

Dans ce chapitre, nous présentons le contexte général de notre recherche. Nous commençons par expliciter dans la section 1.1 nos motivations. A partir de celles-ci, nous présentons respectivement dans les sections 1.2 et 1.3 nos objectifs scientifiques et certains principes que nous adoptons pour les atteindre. Quelques remarques méthodologiques dont nous nous sommes inspirés tout au long de nos travaux sont ensuite énumérées dans la section 1.4. Ensuite, nous décrivons notre démarche dans la section 1.5. Nous finissons ce chapitre en présentant dans la section 1.6 l'organisation de ce manuscrit.

1.1 Motivations

Les systèmes informatiques sont de plus en plus souvent utilisés dans plusieurs secteurs des grandes entreprises, en allant des ordinateurs personnels pour le traitement bureautique jusqu'aux matériels plus sophistiqués, comme des stations de travail, pour le traitement des tâches plus complexes. Par conséquent, nous croyons comme dans [NWM93, Tok93, Hew93] que les environnements de traitement de l'information du futur seront composés de grands réseaux de ressources de calcul hétérogènes, autonomes et distribués, en incluant des ordinateurs, des grosses applications et des grosses données (comme des fichiers et base de données). En particulier, dans [Tok93], l'auteur affirme que dans l'avenir nous n'aurons plus besoin d'avoir des copies locales des programmes, il suffira de demander à un site responsable par un certain service de l'exécuter et d'envoyer ensuite les résultats. Il appelle ces environnements "sociétés d'objets". Une idée similaire est présentée dans [Hew93] sous le nom de "organisations électroniques". Nous pouvons donner comme exemple irréfutable de cette tendance l'importance actuelle du réseau Internet.

Pour créer de tels systèmes, il faut d'abord assurer l'*interconnexion* de ses composantes. Jusqu'à présent, la plupart des systèmes opéraient seuls ("stand-alone"), ne pouvant pas communiquer entre eux. Deux ou plusieurs ressources informatiques sont interconnectées si elles peuvent communiquer entre elles [NWM93].

En plus de l'interconnexion, une deuxième étape à franchir pour la création de tels systèmes de est d'assurer l'*interopérabilité* de ses composantes. Deux ou plusieurs ressources informatiques sont interopérables si elles peuvent interagir pour exécuter une même tâche ensemble [NWM93]. En plus de pouvoir communiquer, il est nécessaire de leur fournir des facilités de traduction de données, car les ressources ne sont pas censées les représenter avec le même formalisme ou au même niveau d'abstraction.

Ces systèmes étant par leur propre nature ouverts, nous croyons qu'une troisième étape à franchir est d'assurer leur *adaptation* aux changements pouvant avoir lieu à cause de l'entrée ou de la sortie dynamique de services.

Enfin, la dernière étape consiste à assurer qu'il y aura effectivement une *coopération* entre ces systèmes. Etant donné qu'un service a un coût associé, nous devons concevoir les moyens à travers desquels un système peut vouloir accepter de coopérer avec d'autres systèmes, puisque dans ce contexte-ci, la coopération ne peut pas être prise en compte comme une hypothèse de départ.

Cette thèse aborde les deux derniers aspects cités ci-dessus, c'est-à-dire, notre intérêt est celui d'étudier comment on peut assurer que des tels systèmes puissent s'adapter et coopérer les uns avec les autres.

1.2 Objectifs scientifiques

Ce travail se place ainsi dans le domaine de l'Intelligence Artificielle Distribuée (IAD), plus particulièrement dans un des ses sous-domaines appelé Systèmes Multi-Agents (SMA).

D'abord, il faut préciser, même si de façon préliminaire, ce que nous entendons par *agent*, *société*, *environnement*, *interaction* et *organisation* puisque ces termes seront utilisés tout au long du manuscrit. Cette décomposition de base est celle de l'équipe MAGMA dans le cadre de laquelle notre travail s'inscrit [Dem95]. Nous affinerons leur définition par la suite.

Etant donné un système, on parlera d'agent pour faire référence à *chacune de ses entités actives*, de société pour désigner *l'ensemble d'agents* et d'environnement pour caractériser *ses entités passives*. Un agent raisonne sur l'environnement, sur les autres agents et décide rationnellement quels sont les buts à poursuivre, les actions à entreprendre, etc. [PLE92]. L'utilisation du terme agent présuppose donc une notion sous-jacente de contrôle [Boi93]. En effet, le terme entité active sert à signaler le fait qu'un agent ne correspond pas aux notions statiques telles que des modules, des ensembles de règles et des bases de connaissances sans avoir associé à de telles notions un contrôle de leur activation. En faisant une analogie très grossière avec les systèmes répartis, un agent correspondrait à un processus, la société à l'ensemble des processus et l'environnement aux entités du monde, n'étant pas d'autres processus, avec lesquelles le système est en relation. Cependant, lorsqu'on parle d'agents et de société d'agents, on s'adresse à un niveau conceptuel plus haut que celui des processus, on se situe dans celui connu par niveau de con-

naissances ("knowledge level"), présenté dans [New82], comme nous l'expliquons dans la suite. Le terme *interaction* désigne les échanges d'informations ayant lieu entre des agents. Enfin, le terme *organisation* fait référence aux contraintes appliquées aux agents au sein d'une société, c'est-à-dire, aux moyens au travers lesquels le concepteur de telles sociétés d'agents ou bien les agents eux-mêmes peuvent garantir que chaque agent voudra et fera ce qui doit être fait et au bon moment.

Dans le cadre de la problématique introduite dans la section précédente, nous nous plaçons dans un contexte d'un *SMA ouvert*, c'est-à-dire, un SMA où des agents peuvent y entrer ou sortir dynamiquement. Par conséquent, comme l'ensemble d'agents appartenant à la société ne peut pas être connu a priori, *l'organisation de la société doit être établie de manière dynamique.* Ainsi, *les agents forment dynamiquement des coalitions* lorsqu'ils ne peuvent pas atteindre leurs buts tout seuls, ce terme désignant cette notion d'organisation créée dynamiquement.

Dans ce contexte, la possibilité de raisonner sur les autres est une fonctionnalité essentielle chez un agent "intelligent". En particulier, puisque des fonctionnalités peuvent être dynamiquement créées ou détruites au sein de la société par l'entrée ou la sortie des nouveaux agents, certains aspects concernant l'*adaptation* de l'agent à cette dynamique de la société qui ne sont pas usuellement abordés dans le cadre d'un système non-ouvert doivent maintenant l'être : un agent doit pouvoir évaluer à tout moment si ses buts sont réalisables et si ses plans sont exécutables. Un plan est dit *exécutable* si toutes les fonctionnalités nécessaires pour l'accomplir sont présentes dans la société. Un but est dit *réalisable* si il existe un plan exécutable qui peut l'atteindre.

En plus, lorsqu'un agent ne peut pas entreprendre ses buts tout seul, il doit pouvoir évaluer la possibilité des autres agents à *coopérer* avec lui, puisque les agents ne sont pas censés a priori s'aider les uns aux autres. Ainsi, en raisonnant sur les buts des autres, un agent peut proposer la formation d'une *coalition* à des agents qu'il croit les plus réceptifs à une telle proposition, et ceux-ci peuvent décider s'ils veulent en faire partie ou non. D'ailleurs, puisque le système est ouvert, ces choix sont en constante évolution : pour atteindre un même but, un agent peut choisir de former une coalition avec des partenaires différents à des moments différents.

Finalement, un agent doit pouvoir *réviser ses croyances* relatives aux autres agents au fur et à mesure qu'il interagit avec eux, car dans un système de ce genre il est hautement improbable que tous les agents aient des informations correctes et complètes les uns sur les autres à tout moment.

Par conséquent, nous défendons la thèse qu'*un agent doit avoir un mécanisme de raisonnement social pour réagir proprement face à ce genre de situations.* Ce mécanisme, faisant partie de l'architecture interne d'un agent, est le responsable par la mise en œuvre des différents aspects énumérés ci-dessus.

Nous appelons *social* un mécanisme qui utilise des *informations sur les autres* pour réaliser ses propres inférences, celles-ci désignant la génération de nouvelles connaissances à partir de ses connaissances courantes. Dans notre contexte, l'exis-

tence d'un tel mécanisme entraîne les hypothèses suivantes :

- un agent doit *représenter explicitement* certaines propriétés concernant les autres agents, ces propriétés pouvant varier dynamiquement. Chaque agent a sa propre représentation des autres, qui lui est *privée*, et dont l'acquisition peut être réalisée par différentes sources, telles que la *perception* (acquisition par l'environnement), la *communication* (acquisition par l'interaction avec d'autres agents) et l'*inférence* ;

- un agent doit *exploiter ces représentations*, lui permettant d'optimiser son comportement vis-à-vis de l'évolution de la société. Cette exploitation lui permet de détecter si à un moment donné ses buts sont réalisables ou non, ses plans sont exécutables ou non, d'évaluer la possibilité des autres agents d'accepter de former une coalition pour atteindre ses buts, et de décider s'il est convenable d'accepter d'en faire partie ;

- un agent doit *réviser ces représentations* lorsqu'il détecte que les informations qu'il possède sur les autres sont incomplètes ou incorrectes. Cette révision est réalisée de façon autonome, c'est-à-dire, sans un contrôle global pré-établi.

Nous avons développé un mécanisme de ce type fondé sur la notion de *dépendance sociale* [CMC92], dorénavant référencée *dépendance* plus simplement. En résumé, nous pouvons dire qu'un agent dépend d'un autre si ce deuxième peut faciliter/empêcher la réalisation d'un but du premier. La notion de dépendance est une notion duale à celle de pouvoir social [Cas90], dans le sens où un agent a du pouvoir sur un autre lorsque ce dernier est dépendant de lui.

Dans cette thèse, nous montrons qu'un modèle fondé sur la notion de dépendance fournit les contributions suivantes :

- d'un point de vue de *modélisation cognitive*, cela introduit un cadre théorique suffisamment riche pour permettre d'analyser et prédire l'occurrence de plusieurs types d'interactions sociales significatives dont la coopération n'est qu'un exemple. En particulier, un tel modèle nous permet de prendre en compte la notion d'adoption de buts, absente dans les approches fondées exclusivement sur la notion d'utilité et qui, à notre avis, explique d'une façon beaucoup plus satisfaisante la procédure de formation de coalitions. Par ailleurs, il nous fournit un moyen d'évaluer la dynamique du pouvoir social des agents appartenant à une société, lorsque certains y entrent ou bien en sortent dynamiquement ;

- en ce qui concerne l'*adaptation* d'un agent, les relations de dépendance lui permettent de savoir à chaque instant si ses buts sont réalisables et si ses plans sont exécutables. Un agent peut ainsi les utiliser pour choisir dynamiquement un but à atteindre et un plan à suivre, en s'assurant que tous

les fonctionnalités nécessaires à l'exécution du plan choisi sont présentes dans la société ;

- en ce qui concerne la *formation de coalitions*, notre modèle introduit la notion de situation de dépendance pour permettre à un agent d'évaluer la possibilité d'un autre agent à adopter ses buts ; de cette façon, le choix de partenaires est fait d'une façon plus efficace, sans supposer que les agents acceptent d'adopter de façon inconditionnelle les buts les uns des autres. Ainsi, un agent peut utiliser cette notion pour choisir un partenaire à qui proposer une coalition lorsqu'il est incapable d'atteindre son but tout seul ou bien pour décider s'il doit accepter de faire partie d'une coalition qui lui est proposée ;

- en ce qui concerne *la révision de croyances*, un tel mécanisme permet de détecter s'il existe une inconsistance entre les représentations que les agents ont les uns des autres. Si nous admettons que le choix des interactions entre des agents est fondé sur le calcul de ses dépendances, c'est justement à partir du déroulement de ces interactions que les agents peuvent détecter si les informations qu'ils ont sur les autres sont à la fois *correctes et complètes*. Dans le cas général des SMA ouverts, la correction et la complétude des informations est une exception, car les agents n'ont qu'une description partielle des autres agents et de l'environnement.

De cette manière, nous montrons que dans le cadre de tels systèmes le raisonnement sur les autres et la révision de croyances sont des traitements imbriqués : un agent utilise ses informations sur les autres pour interagir socialement et de sa part les résultats de cette interaction (si non-réussie) lui permettent de mettre à jour ces informations.

Nous avons conçu deux applications qui exploitent notre modèle, le simulateur DEPNET et le système DEPINT, qui illustrent les points présentés ci-dessus.

Notre mécanisme de raisonnement social peut être analysé selon l'optique du niveau de connaissances [New82], plus spécifiquement d'une de ses extensions appelée niveau coopérative de connaissances ("cooperation knowledge level") dans [Jen92]. Il s'agit de représenter et exploiter au sein d'un agent quelques aspects de l'activité de résolution coopérative de problèmes, en permettant de décrire les diverses types d'interactions sociales, comme par exemple la coopération. Dans cette thèse, nous ne montrons pas *comment un ensemble d'agents résolvent un problème particulier de façon coopérative*. Ce qui nous montrons, c'est *comment un ensemble d'agents se mettent d'accord pour résoudre un problème.*

1.3 Principes adoptés

Dans notre cadre de travail, nous adoptons les quatre principes suivants concernant les agents :

Principe 1. ***Principe de la non-bienveillance*** *: les agents ne sont pas censés a priori s'aider les uns les autres, ils décident de façon autonome s'il acceptent de coopérer ou non avec les autres.*

Principe 2. ***Principe de la sincérité*** *: les agents ne choisissent jamais de façon délibérée de donner une information incorrecte aux autres avec le but de les exploiter ; ils ne communiquent aux autres que ce qu'ils croient eux-mêmes.*

Principe 3. ***Principe de l'auto-connaissance*** *: les agents ont une représentation correcte et complète relative à leurs propres propriétés, c'est-à-dire ils connaissent leurs buts, leurs savoir-faire etc. En ce qui concerne leur représentation des autres, les agents n'ont que des croyances, puisque nous considérons que la source de telles informations peuvent être erronées.*

Principe 4. ***Principe de la consistance*** *: les agents n'ont pas de croyances contradictoires sur les autres. Une fois qu'une inconsistance est détectée, les agents révisent leurs croyances sur les autres afin de rétablir leur consistance.*

1.4 Remarques méthodologiques

Avant d'expliciter comment nous avons exploité chacun de ces aspects énumérés ci-dessus, nous voulons d'abord introduire deux motivations méthodologiques qui nous ont guidés tout au long de notre travail.

1.4.1 Dualité informatique/science cognitive

L'IAD, et donc les SMA, est une discipline qui étudie les phénomènes liés à l'association de plusieurs entités dites intelligentes. Par conséquent, elle a hérité, d'une façon plus spécialisée, du débat concernant les objectifs scientifiques de l'IA. Nous pouvons distinguer deux perspectives scientifiques relativement opposées sur ce sujet [FG87, pages 18-20], représentées dans la figure 1.1 :

- *l'IA est une Science Cognitive*, dont l'objet de la recherche est le raisonnement. Cependant l'IA vérifie ses modèles et ses théories sur la connaissance et sur le raisonnement non pas en expérimentant sur des sujets humains, mais en programmant les calculateurs ;
- *l'IA est une branche de l'Informatique*, où l'ordinateur n'est pas seulement un outil d'investigation, il est l'objet de la recherche elle-même. L'IA cherche plutôt à concevoir des méthodes qui exploitent les ordinateurs pour leur faire réaliser des fonctions nécessitant de l'intelligence. Dans un certain sens, les chercheurs en IA font de l'*ingénierie de la machine intelligente*.

A notre avis, comme dans [Pit92], ces perspectives ne sont nécessairement pas exclusives l'une à l'autre, nous les considérons même comme *complémentaires*. La

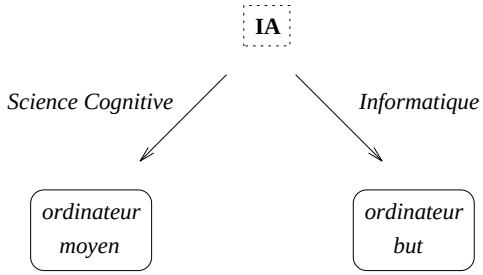


Figure 1.1 : Dualité informatique/science cognitive

source première de ce que nous appelons "intelligence" est sans doute l'homme, et les sciences cognitives ont beaucoup à apporter en ce qui concerne des modèles de comportement intelligents de l'homme. Or ces modèles ne sont souvent pas très bien formalisés ou parfois ils utilisent une formalisation qui ne peut être implémentable. C'est justement là que l'aspect technique prend son importance : c'est à celui-ci d'éventuellement reformuler ces théories en les rendant implémentables. En résultat, nous avons alors un modèle opérationnel de la théorie, qui peut être utilisé soit pour la valider, la tester et la raffiner (l'aspect ordinateur comme moyen) soit comme un élément de base pour le développement des systèmes qui résolvent une certaine classe de problèmes (l'aspect ordinateur comme but).

En ce qui concerne l'IAD, et plus particulièrement les SMA, le même débat est présent [Cas90] :

- la perspective *simulation sociale* (SS), dont le but scientifique est l'étude des interactions sociales "per se ". Il s'agit ici de formaliser, expliquer et analyser les interactions sociales des agents : pourquoi ils interagissent, pourquoi ils coopèrent, quelles sont les propriétés ou attitudes mentales qui doivent être représentées à l'intérieur des agents ;

- la perspective *résolution de problèmes* (RP), dont le but est de développer des techniques, dont la base est constituée des interactions entre agents et de leur organisation, pour résoudre des problèmes de façon coopérative, distribuée. En étendant la définition de [FG87] citée ci-dessus, nous pouvons dire que les chercheurs en IAD font de l'*ingénierie des sociétés de machines intelligentes.*

Comme en IA, nous croyons qu'il est souhaitable de faire une *fusion* de ces deux perspectives. C'est dans cet esprit que nous avons développé deux applications fondées sur le mécanisme de raisonnement social, les systèmes DEPNET et DEPINT, qui servent à illustrer son intérêt respectivement selon les perspectives simulation sociale et résolution de problèmes.

1.4.2 Orientation modèles/systèmes

Une analyse critique de la méthodologie de recherche en IA est présentée dans [Coh91], par rapport à quelques concepts méthodologiques importants, comme par exemple le but de la recherche, les méthodes de modélisation et de validation, etc.

Cette étude se conclut affirmant que la recherche en IA est dominée par deux méthodologies distinctes, appelées *orientée modèles* et *orientée systèmes*. Selon l'auteur, aucune de ces méthodologies n'est suffisante à elle seule pour caractériser les aspects scientifiques et techniques du domaine. En conséquence de ce fait, la plupart de la recherche souffre de certains problèmes méthodologiques chroniques, tels que l'absence d'hypothèses et prédictions, des modèles non-pertinents et des outils analytiques pauvres.

L'auteur propose une méthodologie, appelé MAD (ou triangle écologique) qui combine ce deux méthodologies et qui élimine quelques conditions générant ces problèmes méthodologiques [CGHH89]. Les sommets de ce triangle représentent trois composantes d'un système : son *architecture* (ses structures et traitements internes), son *comportement* (sa capacité à résoudre des problèmes et la façon dont il les résout) et les caractéristiques de l'*environnement* dans lequel il est plongé (dynamique, temps réel, non-prévisible etc.). La méthodologie est composée de sept activités, que nous ne détaillerons pas ici. L'idée principale est de prendre en compte les trois sommets et ses relations pendant la phase de conception d'un système : par exemple, déterminer les relations causales entre l'architecture, son environnement et son comportement.

Par ailleurs, pendant la conception d'un système informatique, on doit faire face toujours à plusieurs *niveaux d'abstraction* différents : descriptions informelles, descriptions formelles, descriptions algorithmiques et implémentations [PLE92, WJ94].

Nous pouvons noter que ces deux dimensions d'analyse sont orthogonales. Dans [CAM94], l'auteur propose une extension qui combine ces deux dimension d'analyse, comme le montre la figure 1.2. Une première dimension concerne les niveaux d'abstraction (description, formalisation et implémentation) et une deuxième correspond au passage des états mentaux (architecture) aux comportements, en passant par l'environnement.

A notre avis, un programme de recherche bien fondé doit prendre en compte cette analyse méthodologique, au moins pour permettre clairement d'être situé dans ce contexte. De plus, il doit si possible attaquer plusieurs de ces points, notamment l'axe représenté dans la figure 1.2 concernant le passage de la description à l'implémentation, en passant par la formalisation.

Nous avons essayé de réaliser ce passage dans le cadre de cette thèse. Notre mécanisme de raisonnement social est à la fois décrit de façon informelle (1), formalisé (4) et implémenté (7), comme nous le montrons respectivement dans les chapitres 4, 5 et 9 de ce manuscrit. Dans le chapitre 4, nous montrons comment, à partir de la connaissance subjective de leurs relations de dépendances et de pouvoir, nous pouvons expliquer diverses formes d'interaction sociale (1→3). D'ailleurs, dans les

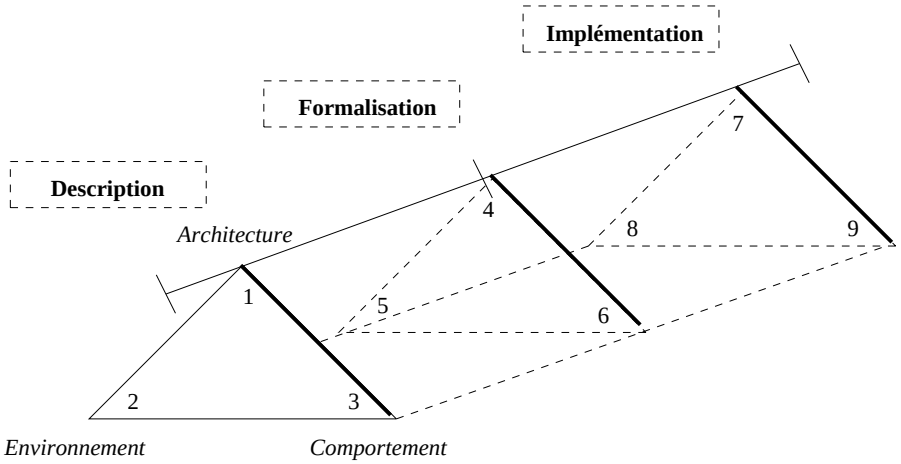


Figure 1.2 : Extension de la méthodologie MAD [CAM94]

chapitres 6, 7 et 8, nous avons aussi spécifié le comportement d'un agent (en ce qui concerne ses divers choix) de façon informelle (3) et formellement (6) et nous avons montré comment notre modèle d'agent est capable de les gérer (1→3 et 4→6). Le système DEPINT illustre que l'implémentation de notre modèle met effectivement en œuvre ces comportements (7→9). En ce qui concerne l'environnement, nous ne l'avons attaqué directement dans cette thèse, il est présent implicitement dans notre contexte de SMA ouverts.

1.5 Démarche suivie

La démarche suivie dans nos travaux est présentée dans la figure 1.3. Elle est composée des activités suivantes :

1. notre hypothèse de départ est la théorie de la dépendance dont l'origine est la psychologie sociale. Cette théorie était initialement présentée de façon informelle dans [Cas90], puis modélisée dans [CMC92] en utilisant le formalisme introduit par Cohen et Levesque [CL87] dans leur travaux sur les intentions. Ce formalisme n'étant pas adapté à une implémentation, nous l'avons reformulé en utilisant une logique de premier ordre, en y ajoutant la notion de plan, absente de cette première formalisation (cf. chapitre 5). Nous avons enrichie la théorie, en introduisant les notions de source d'une dépendance (si elle est localement ou mutuellement reconnue), de situation de but (qui exprime la relation entre un agent et un but) et de situation de dépendance (qui exprime la relation entre deux agents et un but). Ensuite, nous avons implémenté ce modèle, engendrant ce que nous appelons un mécanisme de raisonnement social (cf. chapitre 9). Ce mécanisme construit un réseau de dépendance à partir des informations qu'un agent possède sur

les autres et de ses dépendances calculées. Cette aspect correspond à la *modélisation cognitive* de nos travaux (cf. section 1.2) ;

2. nous avons conçu et implémenté le simulateur DEPNET, avec le but de tester la validité de notre modèle (cf. chapitre 10). En construisant ce système, notre objectif était de fournir un outil informatique aux chercheurs du domaine de la psychologie sociale pour l'analyse du comportement de micro-sociétés, en particulier en démontrant l'intérêt d'utiliser la théorie de la dépendance comme une base pour prédire l'occurrence de plusieurs interactions sociales significatives et pour calculer le pouvoir social des agents (cf. section 4.5.1). En outre, cette phase nous a permis de montrer l'intérêt du mécanisme de raisonnement social selon la perspective SS (cf. section 1.4.1) ;

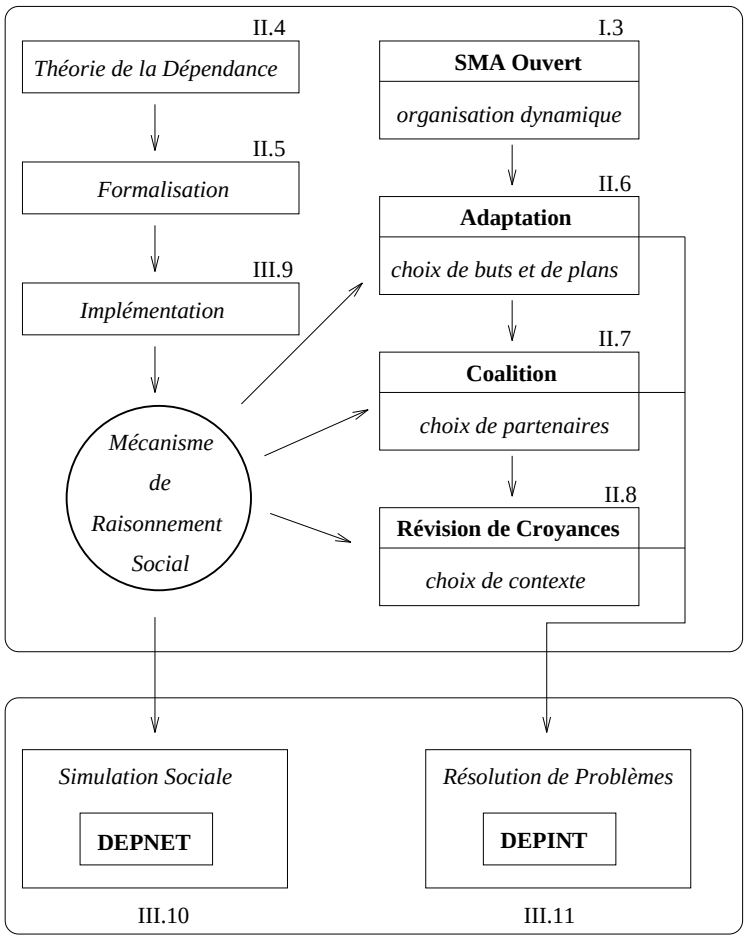


Figure 1.3 : Démarche suivie

3. nous avons adopté le modèle de résolution coopérative de problèmes proposé dans [WJ94] pour concevoir de façon dynamique l'organisation d'une société d'agents (cf. section 3.2). Nous avons ainsi remarqué que notre mécanisme de raisonnement social pouvait être exploité dans le cadre de ce modèle, particulièrement en ce qui concerne la formation de coalitions (cf. section 3.3). Nous avons ensuite proposé un modèle d'agent (cf. section 3.4) et de société (cf. section 3.5) afin de permettre cette exploitation ;

4. nous nous sommes intéressés ensuite aux aspects concernant l'*adaptation* et la *formation de coalitions* (cf. section 1.2). En particulier, nous avons montré qu'un agent pouvait utiliser les relations et les situations de dépendance pour optimiser ses choix de buts (cf. section 6.2), plans (cf. section 6.3) et partenaires (cf. section 7.1). Nous avons proposé un critère de décision pour le choix de partenaires (cf. section 7.1.4) fondé sur la source (cf. section 7.1.2) et la nature (cf. section 7.1.3) des situations de dépendance. Ce critère a été utilisé aussi pour modéliser l'acceptation d'un agent à participer à une coalition (cf. section 7.2). De plus, nous avons analysé l'évolution de ces choix, en démontrant que notre mécanisme de raisonnement social permet à un agent de les changer dans un contexte d'une société ouverte(cf. section 7.3) ;

5. nous avons abordé dans la suite ce que nous avons appelé *révision de croyances* (cf. section 1.2). En utilisant le modèle déductif de croyances, proposé par Konolige [Kon86], nous avons étendu notre formalisme avec des opérateurs modaux (cf. section 8.2). Nous avons prouvé formellement que dans quelques cas où deux agents calculent deux situations de dépendance différentes l'un envers l'autre pour un même but, nous pouvons effectivement prédire une inconsistance au niveau de la société (cf. section 8.3) ;

6. étant donné une inconsistance au niveau de la société, nous voulions étudier un mécanisme de révision de croyances local aux agents. Pour cela, nous avons étudié les travaux accomplis dans le domaine de la révision de croyances, en particulier ceux concernant la fusion de données de divers sources d'information et ceux traitant l'aspect des thèmes d'information (cf. section 8.4). Nous avons caractérisé les divers sources d'information dont un agent peut se servir pour acquérir des informations sur les autres et nous avons proposé un critère de choix, assez restreint, mais qui a servi nos propos (cf. section 8.5) ;

7. nous avons conçu et implémenté le système DEPINT, avec l'objectif de tester nos méthodes de formation de coalition et de révision de croyances (cf. chapitre 11). En outre, cela nous a permis de montrer l'intérêt du mécanisme de raisonnement social selon la perspective RP (cf. section 1.4.1).

1.6 Organisation du manuscrit

Ce manuscrit est composé de douze chapitres et de cinq annexes, organisés de la manière suivante :

- ce chapitre-ci décrit nos motivations scientifiques et guides méthodologiques, la description de la démarche suivie et l'organisation du manuscrit ;

- la partie I, constituée de deux chapitres, est consacrée au domaine des SMA. Dans le chapitre 2, nous spécifions le domaine de recherche de l'IAD, en présentant entre autres son état de l'art, ses principaux axes de recherche et ses deux sous-domaines : la résolution distribuée de problèmes (RDP) et les systèmes multi-agents (SMA). Nous caractérisons leurs différences et donnons des exemples de modèles et systèmes existants. Dans le chapitre 3, nous présentons les approches statiques et dynamiques pour la conception des organisations d'agents et nous justifions qu'un approche dynamique est plus approprié dans notre contexte. Nous présentons une méthode pour former des coalitions, et nous détaillons nos modèles d'agent et de société ouverte qui permettent la mise en œuvre de cette méthode ;

- la partie II est constituée de cinq chapitres. Elle est consacrée à notre mécanisme de raisonnement social, correspondant à la contribution majeure de nos travaux. Dans le chapitre 4, nous introduisons de façon informelle la théorie de la dépendance, issue du domaine de la psychologie sociale. Nous comparons l'approche fondée sur des dépendances avec quelques autres modèles d'interaction sociale, en démontrant des avantages de la première. Notre modèle formel est présenté dans le chapitre 5. Dans ce chapitre, nous définissons ses concepts principaux : les situations de but et les situations de dépendance. L'effet du mécanisme raisonnement social sur l'adaptation des agents à une société ouverte est présenté dans le chapitre 6, où nous détaillons comment un agent peut l'exploiter pour choisir ses buts et plans. Dans le chapitre 7, nous abordons la procédure de formation de coalition, plus particulièrement les critères pour le choix et pour l'acceptation de partenaires. Nous montrons aussi que, en calculant ses relations et situations de dépendance envers des autres agents de la société, les choix et l'acceptation de partenaires peuvent évoluer, dans un cadre d'une société ouverte. Dans le chapitre 8, nous décrivons l'effet de ce mécanisme sur la révision de croyances, en particulier en démontrant comment une inconsistance au niveau de la société peut être détectée par la comparaison des résultats inférés par les mécanismes de raisonnement social de deux agents qui interagissent. Nous proposons un critère de choix de contexte qui permet à un agent de mettre à jour de façon autonome ses informations sur les autres ;

- la partie III est constituée de trois chapitres. Elle est consacrée à la description de l'implémentation de notre modèle et de deux applications qui

l'utilisent. Dans le chapitre 9, nous montrons comment nous avons implémenté le mécanisme de raisonnement social en utilisant une approche orientée objet. Les chapitres 10 et 11 présentent deux applications qui exploitent notre modèle, respectivement le simulateur DEPNET (un outil informatique pour l'analyse du comportement de micro-sociétés) et le système DEPINT (un SMA ouvert qui implémente la méthode de formation de coalitions et de révision de croyances), et qui illustrent respectivement l'intérêt de ce mécanisme selon les perspectives SS et RP introduites dans la section 1.4.1 ;

- nous décrivons dans le chapitre 12 nos conclusions et perspectives de recherche ;

- la dernière partie comprend cinq annexes. Dans l'annexe A, nous présentons les preuves formelles de quelques propriétés de notre modèle. Dans les annexes B et C, nous montrons une autre exploitation possible de celui-ci, que nous n'avons pas abordé directement dans le cadre de cette thèse, et qui fait partie de nos perspectives futures. Il s'agit de l'utiliser comme un outil formel pour décrire et éventuellement concevoir de façon statique des organisations de sociétés d'agents. Dans chacun de ces deux derniers, nous l'appliquons à un système, afin d'illustrer cette utilisation. Nous présentons une liste de abréviations utilisées dans l'annexe D. Enfin, nous proposons dans l'annexe E une liste bibliographique commentée en anglais des principales références sur lesquelles nos travaux se sont appuyés ou bien qui ont été écrits au cours de notre recherche afin d'illustrer ses résultats. Pour chacune d'entre elles, nous indiquons le(s) chapitre(s) de ce manuscrit dans lequel(lesquels) elles ont été utilisées.

Première partie

Systèmes multi-agents

Chapitre 2

Intelligence artificielle distribuée

Ce chapitre a pour objectif de donner une vision générale du domaine de recherche de l'IAD, et de ses deux sous-domaines : la résolution distribuée de problèmes (RDP) et les systèmes multi-agents (SMA). Pour cela, nous présentons une version mise à jour des idées que nous avons initialement présentées dans [SDB92].

Ce chapitre est organisé de la façon suivante. Nous commençons en donnant une caractérisation du domaine dans la section 2.1. Nous finissons cette section en introduisant deux notions de base que nous jugeons essentielles pour situer nos travaux, l'organisation et les interactions entre agents. Ensuite, dans la section 2.2 nous présentons deux points de vue différents pour différencier les deux grands sous-domaines : la RDP et les SMA. Ce dernier sous-domaine, auquel nous nous intéressons plus particulièrement, est décrit plus en détail dans la section 2.3, où nous définissons ce qui est un agent, les divers types d'agents (cognitifs et réactifs) et présentons quelques exemples de modèles et d'applications qui ont été développées pendant les dernières années. Enfin, nous terminons ce chapitre en faisant une synthèse des divers points abordés dans la section 2.4.

2.1 Caractérisation du domaine

Dans les sous-sections suivantes, nous énumérons les aspects principaux du domaine de l'IAD. Nous commençons en présentant la problématique abordée dans la sous-section 2.1.1. Un bref historique est présenté dans la sous-section 2.1.2, où nous citons les travaux pionniers qui ont donné un impulse scientifique à ce domaine. Dans la sous-section 2.1.3, nous listons quelques possibles avantages d'une approche distribuée pour la résolution de problèmes. Les principaux axes de recherche sont énumérés dans la sous-section 2.1.4. Nous terminons cette section en introduisant les notions d'organisation et d'interactions entre agents dans la sous-section 2.1.5.

2.1.1 Problématique abordée

Le domaine de l'IA a connu beaucoup de progrès pendant les années 70. En particulier, diverses méthodes de résolution de problèmes, telles que celles fondées sur la recherche heuristique dans les graphes d'état, et celles fondées sur la représentation et manipulation des connaissances, ont été proposées, étudiées, analysées et validées par le développement de systèmes dans plusieurs domaines d'application. En tant que technique, les systèmes de production ont connu une grande utilisation, en démontrant que la recherche dans le domaine pouvait offrir des résultats concrets, en facilitant le développement des systèmes ayant un intérêt pratique. En outre, comme les axes de recherche étaient très diversifiés, traitant plusieurs aspects différents, le domaine a été divisé en un nombre considérable de sous-domaines, comme la planification, l'apprentissage, et le traitement du langage naturel, chacune ayant ses propres intérêts scientifiques et démarches de développement.

Une caractéristique commune de tels systèmes est l'utilisation d'une métaphore d'intelligence fondée sur le *comportement individuel* d'un être humain, considéré comme la source naturelle de ce que nous appelons "intelligence". Nous ne voulons pas définir ici en quoi consiste l'intelligence, car nous croyons qu'il est très difficile, même impossible, de la caractériser. Pendant ces années, les technologies de réseaux d'ordinateurs n'étaient pas très développés, ce qui rendait difficile la connexion des systèmes développés séparément. De ce fait et de celui de la métaphore utilisée, la plupart des systèmes d'IA présentaient les caractéristiques suivantes :

- *une conception centralisée*, c'est-à-dire que les systèmes étaient exécutés normalement sur une seule machine, et leurs divers traitements étaient exécutés de façon séquentielle. Un mécanisme de contrôle global et centralisé garantissait le bon ordre d'activation de ces traitements ;
- *la non-réutilisabilité*, c'est-à-dire que dans la plupart des cas les divers traitements implémentés au sein de ces systèmes (inférence, communication, etc.) pouvaient difficilement être réutilisables au sein d'un autre système censé résoudre un problème similaire ;
- *la non-ouverture vers l'extérieur*, c'est-à-dire que les systèmes étaient conçus de façon à fonctionner indépendamment de l'existence d'autres systèmes ("stand-alone systems").

Au milieu des années 70 et début des années 80, avec le développement des technologies de réseau, il est devenu techniquement possible et économiquement avantageux de remplacer des gros ordinateurs par des réseaux d'ordinateurs personnels ou des stations de travail. On a connu un grand effort de standardisation pendant ces années pour permettre l'*interconnexion* des systèmes : comment faire communiquer différents systèmes entre eux. Comme exemple de standardisation, nous pouvons citer les modèles OSI ("Open System Interconnection") de l'ISO et le modèle NFS ("Network File System") de Sun Microsystems.

Au même moment, les premières modèles [Len75, AH86] ou systèmes [ACG86, BAF+87, Car87] permettant l'adoption d'une approche distribuée de l'IA sont apparus. C'est la naissance d'un nouveau domaine de recherche, l'IAD, particulièrement d'un de ses sous-domaines, la RDP. La caractéristique principale de cette première génération de systèmes d'IAD était *la distribution* des traitements, qui n'étaient plus activés de façon séquentielle, mais exécutés cette fois-ci de façon concurrente, sur un ensemble de machines connectées par un réseau. Néanmoins, ces systèmes continuaient à être *non-ouverts* et leurs traitements continuaient à être *non-réutilisables*. Leur objectif était de résoudre de façon distribuée un problème bien déterminé. Les techniques pour bien interconnecter et pour faire coopérer un ensemble de sous-systèmes ont été développés, mais généralement ces techniques étaient très dépendantes du modèle algorithmique sous-jacent adopté pour la résolution du problème en question. Dans la plupart des cas, même si les traitements pouvaient être activés de façon concurrente, le système n'avait qu'un seul mécanisme de contrôle global pour les activer. Un exemple classique de ce genre d'approche est le modèle du tableau noir [EHRLR80].

Pendant les années 80 et au début des années 90, des efforts ont été menés dans le sens de permettre la *réutilisabilité* des traitements. Pour cela, diverses méthodes pour implémenter un *contrôle décentralisé* [DM90b] ont été proposées et testées. Les activités de conception des sous-systèmes, des moyens pour les faire communiquer et des moyens pour concevoir leur contrôle et leur coordination deviennent disjointes. Ce nouveau paradigme pose des problèmes intéressants pour l'IAD. Etant donné un système totalement décentralisé, quelles sont les modèles et techniques qui doivent être utilisés pour concevoir ses composantes ? Quelles sont les mécanismes pour lesquels nous pouvons assurer que le comportement global du système soit cohérent ? Comment peut-on concevoir ses sous-systèmes de façon a pouvoir les réutiliser dans des applications similaires ? Ce genre de questions, parmi d'autres, ont donné naissance à un deuxième sous-domaine de l'IAD, les SMA. Un exemple classique de ce genre d'approche est le modèle ARCHON [JW92].

Enfin, plus récemment, un nouveau effort de standardisation est né, cette fois-ci en ce qui concerne l'*interopérabilité* des systèmes : comment faire interagir différents systèmes, conçus séparément, afin d'exécuter une même tâche ensemble. Cette nouvelle tendance met l'accent sur des *services* très complexes disponibles dans un réseau, en présentant des caractéristiques suivantes [CSD93] :

- *distribution* : des fournisseurs et utilisateurs des services peuvent être dispersés géographiquement ;
- *hétérogeneité* : des systèmes informatiques différents peuvent fournir les mêmes services indépendamment du vendeur, de la technologie, de la configuration ;
- *ouverture* : des fournisseurs et utilisateurs des services peuvent entrer ou sortir de l'environnement librement, sans un contrôle global externe.

Comme exemple de ces standards, nous pouvons citer le modèle ODP ("Open Distributed Processing") proposé conjointement par l'ISO ("International Standards Organization") et par la CCITT ("Consultative Committee on International Telephony and Telegraphy") [Tay92] et CORBA ("Common Request Broker Architecture") proposé par le OMG ("Object Management Group") consortium [Rym91]. Une exemple typique de ce genre de système est le réseau WWW ("World Wide Web") [KB95].

C'est intéressant de noter que ces idées n'étaient pas nouvelles, car elles avaient été déjà proposées, sous un autre accent, dans le modèle de *systèmes ouverts* [Hew86, HI91]. Un système ouvert est un système où ses composantes sont conçues séparément, étant en évolution continue. Ces composantes fonctionnent de façon asynchrone et concurrente, avec un contrôle distribué. Certaines composantes peuvent présenter des inconsistances locales, et il n'existe pas a priori un limite globale du système visible à ces composantes.

Ce nouveau paradigme pose des problèmes intéressants pour les SMA. Etant donné un système en constante évolution, censé interopérer avec d'autres systèmes développés séparément, quelles sont les modèles et techniques qui doivent être utilisés pour concevoir ses composantes ? Pourquoi et dans quelles conditions une de ces composantes doit coopérer avec d'autres ? Comment peut-on concevoir dynamiquement l'organisation d'un tel système ? Quels sont les standards que les concepteurs doivent suivre afin de permettre l'interopérabilité de leurs systèmes ? Nous appelons cette classe de systèmes de *SMA ouverts*, et c'est plutôt à ce genre de systèmes que nous nous intéressons dans cette thèse. Nous sommes conscients que d'autres auteurs utilisent l'expression SMA ouvert avec une autre signification : par exemple, dans [CHB94], ce terme désigne des systèmes où des agents peuvent migrer d'une société à l'autre. Néanmoins, nous tenons à utiliser ce terme dans notre contexte, car nous le trouvons très approprié. Comme un possible exemple de cette approche, imaginons un ensemble d'agents conçus séparément pour exploiter l'Internet, afin de chercher des informations scientifiques sur un domaine particulier.

En concluant, le domaine de recherche de l'IAD, né aux milieu des années 70 [BG88], comprend l'étude de modèles et techniques pour résoudre une classe de problèmes dont la distribution, soit physique ou fonctionnelle, est inhérente. En exploitant la possibilité d'interconnecter et d'interopérer des systèmes, elle se consacre à l'étude des modèles et techniques pour concevoir des *systèmes intelligents distribués*, dont l'activation des traitements n'est plus séquentielle. La métaphore d'intelligence utilisée est fondée sur le *comportement social*[1] : il s'agit de chercher comment un collection d'entités indépendantes, dont chacune est appelée *agent* et leur ensemble appelée *société*, peuvent réaliser et coordonner des traitements concurrents de telle sorte que le comportement global du système soit celui désiré par le concepteur.

C'est dans cette perspective de SMA ouverts que s'inscrit notre travail.

[1] Cette terminologie est plutôt utilisée dans le sous-domaine des SMA.

2.1.2 Bref historique

Historiquement, les premiers travaux en IAD sont issus du sous-domaine de la RDP. Au fur et à mesure que quelques modèles provenant de la psychologie et des sciences sociales ou bien de l'éthologie ont été incorporés au domaine, il a eu lieu la naissance du sous-domaine des SMA.

Les premières tentatives de résoudre un problème de façon coopérative remontent aux années 70. Certainement, une de ces tentatives plus connues est le système HEARSAY-II [EHRLR80], un système pour la reconnaissance de la parole qui a introduit le modèle du tableau noir [EM88]. A la même époque, les modèles des acteurs [AH86] et des "beings" [Len75] traitaient des problèmes suivants : le contrôle des ressources partagées, la complexité du contrôle dans les applications distribuées et l'émergence de formes de comportements complexes à partir des interactions très simples.

Plus tard dans les années 80, l'importance de représenter explicitement au sein d'un agent les informations sur les autres, dans un contexte de planification, a été discutée pour la première fois dans [KN80]. L'utilisation des modèles des organisations humaines dans les systèmes censés résoudre un problème de façon coopérative a été proposée dans [Fox81]. En ce qui concerne l'allocation de tâches, le mécanisme de négociation par contrat, utilisant la métaphore économique du libre marché, a été présenté dans [Smi80]. Pendant ces années, la programmation orientée objet est devenue populaire [SB83]. Les premières expérimentations en robotique mobile, en utilisant les robots dits réactifs sont proposées dans [Bro86]. Le modèle des systèmes ouverts, auquel nous imputons une importance fondamentale pour notre recherche, est introduit dans [Hew86]. Finalement, les premiers environnements informatiques pour tester ces idées sont apparus, comme par exemple DVMT [CL83], DPSK [Car87] et MACE [GBH87].

2.1.3 Avantages de l'approche

Une telle approche est désirable pour résoudre des problèmes qui sont à la fois gros et complexes, dont leur résolution utilise des connaissances de différentes domaines, et parfois nécessite un traitement géographiquement distribué. Comme exemple de cette classe de problèmes, nous pouvons citer le contrôle de trafic aérien [SC81] et la monitoration de véhicules [DLC87]. Du point de vue du calcul, les avantages d'une approche distribuée pour traiter cette classe de problèmes sont les suivantes : le contrôle de la complexité, l'amélioration de la réponse dégradée, le support à l'évolution, la possibilité d'utiliser des machines moins coûteuses et la possibilité de dupliquer des traitements lorsqu'ils doivent être activés dans divers régions différentes de l'espace d'entrées [Cha81].

Dans notre travail, nous nous intéressons plutôt à l'aspect concernant l'évolution du système. Nous montrons comment un mécanisme de raisonnement social peut permettre à un agent d'*adapter son comportement* aux sein d'un SMA ouvert, où des agents peuvent entrer et sortir dynamiquement de la société.

2.1.4 Principaux axes de recherche

La recherche en IAD peut être divisée en 5 axes de recherche [BG88, chapitre 1] : (i) description, décomposition et allocation de tâches, (ii) interaction, langage et communication, (iii) coordination, contrôle et comportement cohérent, (iv) conflit et incertitude et (v) langages et environnements de programmation.

Dans ce travail, nous nous concentrons particulièrement dans le premier et quatrième aspects cités ci-dessus. En ce qui concerne l'allocation dynamique de tâches, nous montrons comment un mécanisme de raisonnement social peut permettre à un agent d'influencer d'autres agents à entreprendre des actions qui lui l'intéressent pour atteindre ses propres buts. En particulier, nous proposons l'utilisation d'un modèle ascendant ("bottom-up") pour la formation de coalitions, ce terme étant défini avec plus de précision dans la section 3.2. En ce qui concerne le conflit et l'incertitude, nous montrons aussi qu'un tel mécanisme permet aux agents de détecter des éventuelles informations incomplètes ou incorrectes qu'ils ont les uns sur les autres.

2.1.5 Organisation et interaction entre agents

Pour pouvoir mieux caractériser les différents travaux dans le domaine de l'IAD, nous avons besoin d'introduire les deux notions suivantes [Dem94] :

- *l'organisation des agents* : jusqu'à présent, il n'y a pas ni une approche ni une définition unique pour ce terme dans la communauté IAD. Néanmoins, cette notion exprime les moyens au travers desquels on peut garantir que chaque agent fera ce qui doit être fait et dans le bon moment. Plusieurs alternatives de représentation de cette notion ont été proposés : structures d'autorité et leurs attributs, structures et leurs tactiques de contrôle associés, contraintes sur les liens de communication et sur le comportement des agents, modèles basés sur des paires du type (connaissance, action) etc. Parfois, cette notion est désignée par des termes différents. Par exemple, dans [Car87], ce que nous appelons organisation est appelé contrôle au niveau système ;

- *l'interaction entre agents* : dans la communauté IAD, et particulièrement en SMA, on associe une signification différente au terme communication que celle utilisée dans la communauté des systèmes répartis. Par exemple, on adresse des problèmes tels que la définition de primitives de communication qui puissent exprimer la nature de travail coopératif et leurs structurations possibles au sein de protocoles d'utilisation générale, non nécessairement liés à une application particulière. Nous utilisons ainsi le terme interaction pour faire référence à ces aspects.

Nous montrons dans la section suivante que les moyens par lesquels les agents, leur organisation et leurs interactions sont conçues permettent de différencier les deux sous-domaines de l'IAD.

2.2 Résolution distribuée de problèmes et systèmes multi-agents

Comme nous avons déjà discuté dans la section 2.1, soit pour des raisons historiques, soit par des motivations scientifiques diverses, le domaine de l'IAD peut être divisé en deux sous-domaines : la RDP et les SMA. Dans les sous-sections 2.2.1 et 2.2.2, nous décrivons deux points de vue différents concernant les différences entre la RDP et les SMA.

2.2.1 Première comparaison entre les sous-domaines

Cette première comparaison entre les sous-domaines, inspirée dans [Dem94], est plutôt liée à notre avis à l'aspect *réutilisabilité du logiciel.*

Résolution distribuée de problèmes

Le point de départ de l'approche RDP est un *problème* précis devant être résolu, dont la démarche de résolution présente les caractéristiques suivantes :

- le problème est résolu par un ensemble d'agents, physiquement distribués dans plusieurs machines interconnectées par un réseau. Les agents sont conçus pour résoudre un problème particulier ;
- une organisation est conçue pour contraindre le comportement des agents. Cette organisation est définie complètement pendant la phase de conception du système ;
- l'interaction entre les agents est faite soit par l'échange de messages, soit par le partage d'informations communes. La structuration de ces échanges est définie complètement pendant la phase de conception du système, étant très liée au modèle algorithmique sous-jacent (par exemple, le tableau noir) et au problème que le système doit résoudre ;
- les agents sont exécutés de façon concurrente, pour augmenter la vitesse de la résolution ;
- les agents coopèrent en partageant diverses parties du problème (des sous-problèmes ou des tâches), ou peuvent appliquer des différentes techniques de résolution pour une même tâche ;
- il existe un *contrôle global*, dans la plupart des cas implicite aux agents, qui garantit la cohérence du comportement global du système, selon l'organisation prévue au départ. Ce contrôle peut être implémenté soit de façon centralisée (par la création d'un agent responsable par la gestion du système) soit de façon distribuée.

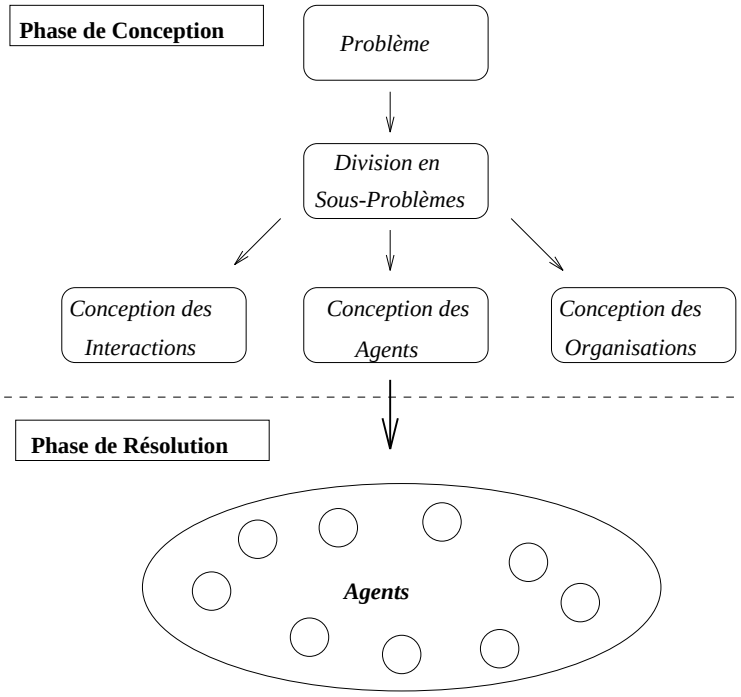


Figure 2.1 : Approche résolution distribuée de problèmes

L'approche RDP est présentée dans la figure 2.1. Du point de vue de conception du système, les agents n'existent pas a priori : leur conception, ainsi que celle de leur organisation et de leurs interactions est due à l'existence d'un problème que le système doit résoudre. La réutilisabilité des agents, de leurs interactions et de leur organisation n'est donc pas possible. En quelque sorte, nous pouvons considérer les systèmes de RDP comme une croisement des techniques des domaines des systèmes répartis et de l'IA, qui appliquent des techniques de coordination et synchronisation développées par le premier domaine [Fer93] pour intégrer des systèmes développés selon les démarches de conception du deuxième.

Systèmes multi-agents

La problématique scientifique abordée par les SMA est différente. Les SMA abordent l'étude de l'activité des *agents autonomes* dans un univers multi-agent. Par autonome, nous voulons dire ici que les agents ont une existence propre, indépendamment de l'existence des autres agents [DM90b][2]. Comme il n'existe pas a priori un problème que le système est censé résoudre, il s'agit d'étudier des modèles généraux à partir desquels nous pouvons concevoir les agents (considérés

[2]Nous analysons plus en détail les différentes significations associées au terme autonomie dans la section 4.2.

ici comme capables de réaliser certains traitements[3]), les organisations et les interactions, de façon à pouvoir les instancier lorsqu'on doit résoudre un problème particulier. Autrement dit, on conçoit les moyens au travers desquels on peut assurer que ces agents vont coopérer les uns avec les autres pour résoudre un certain problème, lorsque celui-ci est posé au système.

Du point de vue de la conception d'un tel système, nous avons les caractéristiques suivantes :

- les agents sont conçus indépendamment d'un problème particulier devant être résolu, ceux-ci permettent la mise en œuvre d'une spécification fonctionnelle : il s'agit de concevoir un agent capable de réaliser un certain traitement et non d'un agent capable de réaliser ce traitement exclusivement au sein d'une application cible particulière ;

- la conception des interactions entre agents est elle-aussi réalisée indépendamment d'une application cible particulière. En particulier, nous pouvons développer des *protocoles d'interaction* assez génériques, qui peuvent être réutilisés dans d'autres applications similaires. Un protocole doit néanmoins être instancié pour pouvoir être utilisé dans une application ;

- la même discussion concernant les protocoles d'interaction s'applique à la conception des organisations. Normalement, on distingue les fonctionnalités nécessaires à une résolution particulière des agents qui vont effectivement les effectuer ;

- pendant la phase de résolution du système, les agents exploitent leurs représentations locales des protocoles d'interaction et des organisations. De cette manière, il n'existe pas de contrôle global du système, celui est *totalement décentralisé* chez les agents [DM90a].

L'approche SMA est présentée dans la figure 2.2. Du point de vue de conception du système, les agents, les organisations et les interactions sont conçus indépendamment d'un problème particulier devant être résolu. Il devient donc possible de réutiliser ces composantes pour concevoir des applications similaires. Les agents, dynamiquement, vont instancier les organisations et interactions lorsqu'un problème est posé au système. En quelque sorte, certaines propriétés globales du système qui étaient totalement développées par le concepteur et liés aux modèles algorithmiques sous-jacents dans une approche RDP sont maintenant *explicites et exploitables* par les agents eux-mêmes.

Remarques

Notre travail se positionne dans le domaine des SMA avec cette perspective. Nous ne nous intéressons pas cependant dans cette thèse à proposer un modèle

[3]Nous proposons dans la section 2.3.1 une définition d'agent qui illustre bien cet aspect.

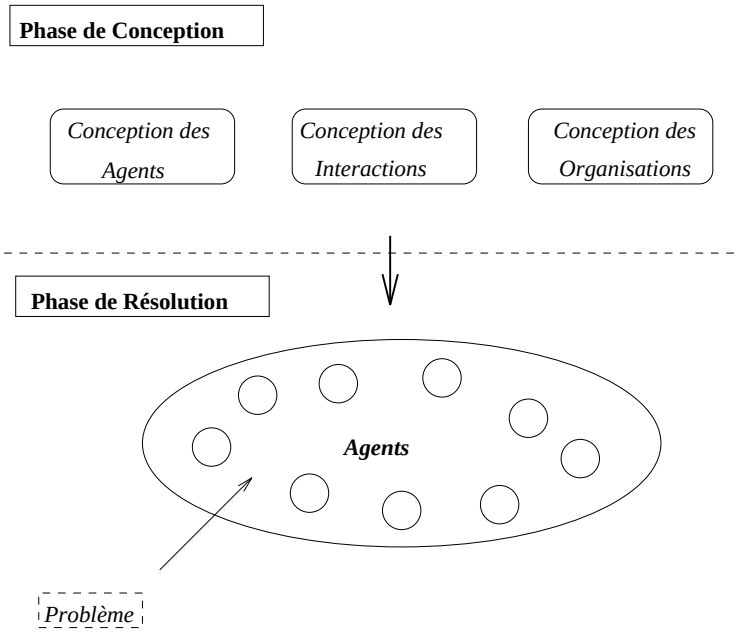


Figure 2.2 : Approche système multi-agents

détaillé d'interactions entre agents. La base théorique utilisée dans la plupart des travaux dans ce domaine est la théorie des actes de langage [Sea69], comme par exemple dans [CD90, Sin91, CW92]. Néanmoins, nous avons participé à quelques travaux sur ce sujet décrits dans [PBS93], dont l'évolution est présentée dans [DBK94a, Dem95]. Nous les expliciterons dans le chapitre 11 lorsque nous décrirons les protocoles d'interaction utilisés dans le système DEPINT.

Ce qui nous intéresse plutôt c'est de proposer un modèle pour la création dynamique d'organisations entre agents, selon nos objectifs présentés dans la section 1.2. Les aspect généraux sur un modèle de ce type sont présentés dans la chapitre 3 suivant, et l'utilisation de la notion de dépendance pour les concevoir est détaillée dans la partie II du manuscrit.

Enfin, il faut noter que selon cette perspective d'analyse, l'hypothèse de bienveillance n'est pas du tout abordée. D'ailleurs, nous croyons plutôt qu'elle est considérée implicitement comme valable. Néanmoins, cette notion peut aussi être utilisée pour différencier ces sous-domaines, comme nous le montrons dans la suite.

2.2.2 Deuxième comparaison entre les sous-domaines

Cette deuxième comparaison entre les sous-domaines, proposée dans [DR94], est plutôt liée à notre avis à l'aspect *bienveillance des agents*.

Les trois relations possibles

Dans [DR94], trois relations possibles entre la RDP et les SMA sont présentées :

- *la RDP comme sous-ensemble des SMA* : nous pouvons considérer la RDP comme un cas particulier des SMA où quelques hypothèses sont valables :
  - ⋆ *l'hypothèse de bienveillance* : des agents veulent a priori s'aider les uns les autres, ils ne se posent jamais la question de pourquoi ils doivent coopérer avec des autres agents[4] ;
  - ⋆ *l'hypothèse de but commun* : pour raffiner l'hypothèse précédente, nous pouvons considérer que tous les agents ont a priori un même but commun ;
  - ⋆ *l'hypothèse de la conception centralisé* : cette notion est plus récente, et concerne le fait qu'un tel système est conçu par un (groupe de) concepteur(s), dont l'objectif est de résoudre un problème déterminé ;
- *les SMA comme base pour la RDP* : selon ce point de vue, les SMA étudient certaines propriétés internes aux agents. Par exemple, la décision de coopérer ou non et l'honnêteté des agents pour communiquer certaines données. Ces propriétés, implémentés de façon décentralisée chez les agents, peuvent fournir une base SMA pour la conception des interactions et organisations des agents au sein d'un système devant résoudre un problème particulier, utilisant des techniques de la RDP ;
- *la RDP et les SMA comme domaines de recherche complémentaires* : même si nous considérons les deux points de vue précédents, parfois il est impossible à un observateur extérieur de discerner s'il s'agit d'un système conçu selon l'approche RDP ou l'approche SMA. Ce fait montre qu'il est peut être plus convenable de distinguer les sous-domaines non pas par l'observation du comportement global du système mais par les motivations scientifiques et techniques des concepteurs et par la nature des expérimentations envisagées.

Les auteurs proposent la table 2.1 pour caractériser les centres d'intérêt et les démarches scientifiques de ces deux sous-domaines. Selon leur analyse, l'approche RDP a comme centre d'intérêt l'obtention de certaines propriétés au niveau système (telles que l'efficacité et la robustesse) à partir des agents pré-définis, en considérant un environnement variable. Pour sa part, l'approche SMA s'intéresse plutôt à l'obtention de certaines propriétés au niveau système à partir d'agents dont les propriétés individuelles peuvent changer, en considérant quelques classes d'environnements pré-établis. Dans aucun cas, les propriétés du système sont variables, ce qui les a amené à définir une troisième ligne dans ce tableau.

[4] Nous caractérisons cette notion de bienveillance sur une autre perspective dans le chapitre 4.

Sous-Domaine	Agent	Environnement	Système
SMA	variable	fixé	fixé (interne)
RDP	fixé	variable	fixé (externe)
?	fixé	fixé	variable

Table 2.1 : Comparaison entre les approches RDP et SMA [DR94]

Remarques

Nous croyons d'abord que les aspects énumérés ci-dessus expriment bien les différences entre les deux approches, de façon complémentaire à ceux que nous avons présentés dans la section précédente. Cette analyse est bien fondée, et mérite d'être saluée comme un premier essai pour caractériser plus proprement la différence entre la RDP et les SMA. En particulier, elle prend en compte la notion de bienveillance, absente de notre première analyse, et que nous considérons comme fondamentale, selon notre principe 1.

En ce qui concerne la table 2.1, tandis que l'analyse de la RDP est assez correcte, à notre avis elle est un peu restrictive dans le cas des SMA, pour les trois raisons suivantes :

- nous ne sommes pas d'accord sur le fait que le but scientifique de l'approche SMA est toujours celui d'obtenir quelques propriétés pré-établies au niveau d'un système. Par exemple, dans une perspective SS, introduite dans la section 1.4.1, il n'existe aucune notion pré-établie d'un comportement global atteint par un système. A notre avis, les auteurs adoptent donc une vision réductrice du domaine des SMA ;

- nous concevons les SMA comme des systèmes ouverts tels que nous les avons définis et cet aspect est absent de ce tableau ;

- l'aspect concernant la réutilisabilité de systèmes telle que nous l'avons exprimé dans la section précédente n'est guère considéré dans cette analyse.

2.2.3 Approche Suivie

Nous croyons que ces deux points de vue sont en quelque sorte complémentaires l'un à l'autre. Ainsi, nous adoptons dans notre démarche quelques idées des deux que nous croyons essentiels pour atteindre nos objectifs. D'une part, dès le début, nous avons adopté le principe de non-bienveillance (P1). D'autre part, notre mécanisme de raisonnement social est conçu comme étant un parmi plusieurs mécanismes internes d'un agent. Ainsi, il est censé être *réutilisable* dans un contexte plus large, car nous ne considérons pas que les agents doivent être homogènes en ce qui concerne leur constitution interne. Nous pouvons considérer notre mécanisme comme la mise en œuvre d'une méthode décentralisée pour la

formation dynamique d'organisations. Nous exploitons ces deux aspects plus en détail dans le chapitre 3 suivant.

2.3 Modèles d'agents en univers multi-agents

Dans cette section, nous présentons deux définitions possibles d'agent. Nous montrons aussi qu'il existe deux approches complémentaires pour les modéliser, en ce qui concerne leur granularité. Enfin, nous présentons quelques exemples d'applications qui ont été développés dans le cadre de ce domaine de recherche.

2.3.1 Définition d'un agent

Actuellement, nous ne trouvons pas dans la littérature une définition d'agent globalement acceptée par la communauté de recherche. Nous adoptons ici deux définitions, à notre avis complémentaires.

Une première définition très générale est proposé dans [FG91], basée sur ses traitements internes :

"*On appelle agent une entité réelle ou abstraite qui est capable d'agir sur elle-même et son environnement, qui dispose d'une représentation partielle ce cet environnement, qui, dans un univers multi-agent, peut communiquer avec d'autres agents, et dont le comportement est la conséquence de ces observations, de sa connaissance et des interactions avec des autres agents.*"

Nous pouvons noter que cette définition est très générale. En particulier, les types des interactions entre des agents ne sont pas complètement définis, ainsi que leur granularité. Par ailleurs, la notion d'autonomie est absente de cette définition. Comme nous le verrons plus tard, nous l'introduirons dans la section 4.2 en nous inspirant de [DM90b] et [Cas90].

En considérant un autre aspect, celui de l'identité d'un agent, nous pouvons donner une deuxième définition, inspirée par [Gas92a] :

"*Un agent est une entité à laquelle nous pouvons associer une identité unique, et qui est capable de fournir des traitements de calcul. Un agent peut être considéré comme étant un moyen qui produit un certain nombre d'actions à partir des connaissances et mécanismes internes qui lui sont propres.*"

Selon cette définition, un agent est une entité *insécable*. Nous croyons que cette définition est complémentaire à la première présentée ci-dessus. A notre avis, elle est très bien adaptée à l'aspect réutilisabilité du logiciel que nous avons abordé dans la section 2.2.1.

Nous croyons que ces définitions sont complémentaires, car elles adressent des aspects différents. Nous en allons tenir compte de ces deux définitions lorsque nous définissons notre modèle d'agent dans le chapitre 3.

2.3.2 Agents cognitifs et agents réactifs

Comme nous avons cité dans la section 2.1, la métaphore d'intelligence utilisée dans les SMA est celle du *comportement social*. Un comportement social "intelligent" peut apparaître dans des sociétés où ses membres sont de agents très simples, à fin grain ou dans des sociétés où ses membres sont des entités complexes, à gros grain. Nous appelons ces premiers *agents réactifs* et ces derniers *agents cognitifs*.

Agents réactifs

Des agents réactifs sont fondés sur des modèles d'organisations *biologiques* ou *éthologiques*, comme les sociétés de fourmis. Dans ces sociétés, nous pouvons observer l'émergence de quelques comportements "intelligents", rendus "évidents" au niveau de la société, mais non pas au niveau agent. Par exemple, une colonie de fourmis semble avoir un comportement intelligent en recherchant de la nourriture, même si nous ne considérons pas une fourmi toute seule comme une entité "intelligente".

Un agent réactif fonctionne selon un modèle *stimulus-réponse*. Sa représentation de l'environnement et des autres agents n'est pas explicite, il n'est capable ni de se rappeler de ce qu'il est arrivé dans le passé, ni de planifier ses actions dans le futur. Chaque agent prend connaissance des actions et des comportements des autres en percevant les modifications dans l'environnement. Il n'y a pas de communication directe entre des agents. Normalement, les sociétés d'agents réactifs sont composés par un grande nombre d'agents, dans l'ordre de milliers.

Nous ne nous intéressons pas dans cette thèse à ce type d'agents et de sociétés[5]. Cependant, beaucoup de chercheurs ont travaillé selon cette perspective, en proposant des modèles plutôt généraux comme l'architecture de subsumption [Bro86], les actions situées [RK86, WG95], le modèle PACO [Dem91, Dem93], l'eco-résolution [FJ91, DD92], l'éthomodélisation [Dro93], et les systèmes téleo-réactifs [Nil94].

Agents cognitifs

Des agents cognitifs sont fondés sur des modèles d'*organisations sociales humaines*, telles que les groupes, les hiérarchies et les marchés. Les agents ont *une représentation explicite de l'environnement et des autres agents*. Au contraire des agents réactifs, ils ont une mémoire, c'est-à-dire qu'ils peuvent raisonner sur le passé et planifier leurs actions dans le futur. En ce qui concerne leurs interactions, ils communiquent directement, par l'envoi de messages. Ainsi, selon notre point de vue, leurs activités de perception et de communication sont distinctes, contrairement aux agents réactifs. Typiquement, le nombre d'agents dans une société de ce type n'est pas très grand, de l'ordre d'une ou deux dizaines.

[5]Pour cette raison, nous utilisons le terme agent tout court comme synonyme d'agent cognitif dans tout le reste de ce manuscrit. Quand nous aurons besoin de faire référence aux agents réactifs, nous les désignerons explicitement.

Il existe plusieurs modèles d'agents cognitifs dans la littérature. Quelques uns s'intéressent au niveau purement formel, dont l'objectif principal est celui de caractériser et modéliser les attitudes mentales d'un agent ou d'un groupe d'agents, telles que les buts [Wai94b], les intentions et les engagements [CL87, CL90, KP93, LCN90], les plans [GS90, Pol86, Pol90, RGS92] et les conventions [Jen93]. Une modélisation théorique complète d'un agent est présenté dans [Wer89, Woo92]. Quelques autres s'adressent au niveau algorithmique, en proposant des architectures permettant d'implémenter les divers mécanismes internes d'un agent [Lev90, OC91, BS92, Boi93, BD94c], et finalement nous pouvons trouver une fusion des deux approches, comme dans [Gas94] et aussi dans notre travail, comme nous le verrons dans la suite.

Agents mixtes

La différence entre des agents réactifs et des agents cognitifs peut être expliquée par le compromis efficacité/complexité. Du fait que leur comportement est totalement "câblé", les agents réactifs ont une performance plus efficace en ce qui concerne le temps de réponse du système. D'ailleurs, les premiers systèmes multi-agents réactifs sont issus des domaines tels que la robotique et les systèmes temps-réel, où le temps de réponse joue un rôle crucial. Néanmoins, l'absence de délibération peut limiter les comportements de tels systèmes. Nous sommes d'accord avec [BD94a], où on affirme que le problème majeur n'est pas de savoir s'il doit y avoir ou non de délibération au sein d'un système, la question est de comment nous pouvons la contrôler. Dans ce travail-là, ainsi que dans [RW91], on propose quelques mécanismes de méta-raisonnement, fondés sur l'utilité espérée d'un traitement (cf. section 4.1.2).

Dans le domaine des SMA, nous pouvons aussi observer quelques propositions de modèles d'agents mixtes, comme ceux décrits dans [CGHH89, Fer92, BD94d, OD94]. Dans [Rod94], l'auteur présente un excellent état de l'art sur ce sujet.

2.3.3 Exemples d'applications

En considérant les SMA réactifs, nous pouvons trouver des exemples de systèmes dans divers domaines d'application telles que la généralisation cartographique [BDA95], la simulation de particules [BJV94], la simulation de véhicules autonomes [FD90, HDL92] et la robotique mobile [Ste90] ou de manipulation [OPP94].

De même, pour les SMA cognitifs, les domaines d'application sont aussi nombreux, nous pouvons citer la reconnaissance du langage naturel [Ste93], les tuteurs intelligents [CGR92], la vision par ordinateur [BD94c], la robotique [OCR91], des systèmes d'automation industrielle [JW92] et les télécommunications [KDEQ95].

Finalement, ces dernières années, nous pouvons trouver des travaux centrés plutôt sur un autre aspect : des agents comme étant des assistants personnels d'un être humain, en les aidant à réaliser des tâches telles que la recherche d'informa-

tions [ELS92, EW94], le filtrage d'informations [Mae94] et la détermination de l'emploi du temps [MCF+94].

2.4 Synthèse

Dans ce chapitre, nous avons montré que la façon dont l'organisation des agents est conçue est très différente entre les deux approches RDP et SMA. Tandis que dans les systèmes de RDP, celle-ci est souvent implicite et liée aux modèles algorithmiques sous-jacents, dans les SMA, celle-ci devient un objet d'étude de première importance, censé être réutilisable dans un contexte d'applications plus large. Nous avons montré aussi que dans un niveau d'abstraction plus haut, l'hypothèse de bienveillance joue un rôle aussi important pour caractériser la différence entre ces deux sous-domaines.

Notre travail s'inscrit ainsi dans un perspective de SMA ouverts, comme décrit dans la section 2.1. Dans le chapitre suivant, nous décrivons comment des agents peuvent concevoir eux-mêmes, de façon dynamique et décentralisée, l'organisation d'une société. Nous proposons l'utilisation d'un mécanisme de raisonnement social pour cette tâche, celui-ci censé être *réutilisable* dans plusieurs modèles d'agents différents, *en ne supposant pas qu'ils soient bienveillants*, comme nous l'avons exprimé dans la section 2.2.3.

Chapitre 3

Conception des organisations

Dans ce chapitre, nous décrivons notre approche pour concevoir l'organisation d'un SMA ouvert. Cette conception est réalisée de façon dynamique et décentralisée. La méthode que nous proposons est basée sur la formation de coalitions entre agents. Dans cette méthode, nous ne supposons pas que les agents soient toujours bienveillants.

Ce chapitre est organisé de la façon suivante. Dans la section 3.1, nous présentons deux types d'approches pour concevoir des organisations : l'approche statique et l'approche dynamique. Nous justifions que dans notre contexte, l'approche dynamique est plus appropriée, et nous décrivons une méthode de formation dynamique de coalitions dans la section 3.2. Dans la section 3.3, nous montrons qu'un mécanisme de raisonnement social tel que nous le proposons est essentiel pour permettre aux agents de former dynamiquement les coalitions entre eux. Enfin, nous terminons ce chapitre en présentant respectivement dans les sections 3.4 et 3.5 nos modèles d'agent et de société qui permettent la mise en œuvre de la méthode proposée.

3.1 Approche statique et approche dynamique

A partir des études empiriques à grande échelle dans les organisations humaines, deux propriétés concernant ces dernières ont été prouvées [Gal73] : (i) étant donnée un problème, il n'existe pas une organisation unique qui optimise sa résolution et (ii) les organisations ne sont pas équivalentes par rapport à leur efficacité de résoudre un problème. Dans ce travail, l'auteur propose une théorie appelée *théorie des contingences* pour permettre à une organisation de s'adapter aux incertitudes qui peuvent avoir lieu pendant l'exécution de diverses tâches. Dans ce contexte-là, une incertitude exprime le fait que pendant l'exécution d'une tâche, on peut avoir besoin d'informations qui n'étaient pas disponibles au moment de sa planification. Autrement dit, des liens d'autorité et de communication sont crées dynamiquement pour faire face à ce genre de problèmes.

Des études plus récentes montrent [HH94] qu'en effet dans les entreprises per-

formantes coexistent deux types d'organisations : (i) celle qui est formelle, dont l'objectif est de guider ses membres à une attitude de collaboration et (ii) celle qui est informelle, où la collaboration entre des individus est faite de façon spontanée. Pour faire référence à cette dernière, on parle de l'émergence[1] des *communautés de la pratique* ("communities of practice").

De façon analogue, nous pouvons trouver dans la littérature de l'IAD deux types d'approches pour la conception des organisations d'une société d'agents :

- l'approche *statique* : selon cet approche, les liens d'autorité et de communication entre des agents, que l'on connaît tous a priori, sont complètement définis pendant la phase de conception du système. Ils servent à établir un moyen de contrôle global de la société, en la conduisant à un comportement censé résoudre un problème. Un exemple de cette approche est le système MAVI [BD94c]. Quelques travaux se consacrent à l'étude de la dynamique de ces organisations, c'est-à-dire comment celles-ci peuvent être changés pendant la phase de résolution de problème afin d'augmenter l'efficacité globale du système [MIT90, IGY92, LS95] ;

- l'approche *dynamique* : selon cette approche, les liens d'autorité et de communication ne sont pas pré-établis entre les agents. Ces liens sont crées de façon dynamique, lorsque les agents cherchent à atteindre leurs propres buts. Un exemple de cette approche sont les divers modèles d'interaction basés sur la théorie des jeux [RZ94].

En ce qui concerne nos objectifs scientifiques présentés dans le chapitre 1, une approche statique pour la conception des organisations ne nous convient pas pour les raisons suivantes :

- comme nous nous plaçons dans un contexte d'un SMA ouvert, nous ne pouvons définir a priori les savoir-faire des agents qui font partie de la société ;

- nous n'adoptons pas l'hypothèse de bienveillance, nous ne considérons pas que les agents ont un but commun a priori, et donc l'approche statique ne peut pas être appliquée, car elle suppose que les agents vont contraindre leur comportement selon une organisation pré-établie de la société ;

- nous voulons utiliser notre modèle de raisonnement social aussi dans une perspective SS, et nous montrons dans la section 4.1.1 que les modèles

[1] Nous voulons faire ici une petite remarque d'ordre terminologique. D'autres auteurs sans doute désigneraient ce que nous appelons l'approche dynamique par le terme "émergence". Nous évitons explicitement l'utilisation de ce terme, car à notre avis une notion sous-jacente de non-controlabilité ou bien de "découverte" lui est implicitement associé. Nous le croyons plus approprié dans une perspective SS, ou l'observateur externe d'un système se "surprend" en voyant que quelques organisations ont lieu dynamiquement. D'ailleurs, c'est pour cette raison que l'utilisation du terme dans le domaine des SMA est plutôt associé à systèmes réactifs [LS95, pages 30-31]. Comme notre approche est dirigée soit à une perspective SS soit à une perspective RP, nous préférons utiliser le terme formation dynamique pour la caractériser.

d'interaction sociale basés sur des structures organisationelles présentent certains inconvénients à cet égard.

Par conséquent, nous ne nous intéressons pas dans cette thèse à présenter l'approche statique pour la conception des organisations. Une discussion plus pointue sur ce sujet peut être trouvée dans [Gal73, Fox81, Mal87, MIT90, Bou92, Gas92b, IGY92, Car92, CFO+93, YM93, CCC93]. Un excellent état de l'art sur ce sujet est présenté dans [LS95].

3.2 Coalitions entre agents

Dans notre contexte, nous utilisons ainsi une approche dynamique pour la conception des organisations. Nous supposons que les agents forment dynamiquement des *coalitions* quand ils croient qu'il est nécessaire de travailler de façon coopérative pour atteindre un certain but. Nous nous inspirons dans le modèle formel de résolution coopérative de problèmes proposé dans [WJ94], qui comprend les quatre phases suivantes, représentées dans la figure 3.1 :

1. *reconnaissance du potentiel de coopération* : une résolution de problème coopérative commence quand un agent reconnaît le potentiel pour une action coopérative ; cette reconnaissance est due au fait que l'agent a un but à atteindre lequel il ne peut pas atteindre tout seul ou bien parce qu'il préfère l'atteindre en coopérant avec d'autres agents ;

2. *formation d'une coalition* : pendant cette phase, l'agent qui a reconnu le potentiel de coopération demande de l'aide aux autres, en communiquant avec eux. Si cette phase est bien réussie, nous aurons comme résultat la formation d'une coalition, c'est-à-dire la formation d'un groupe d'agents ayant un engagement commun pour une certaine action collective ;

3. *formation de plan* : pendant cette phase, les agents essaient de négocier un plan commun pour atteindre le but désiré. Un mécanisme de négociation permet aux agents de montrer leurs préférences par rapport au plan devant être choisi ;

4. *action de la coalition* : après avoir été choisi, le plan est exécuté par des agents, qui maintiennent une relation très stricte entre eux. Des conventions sont suivies pour assurer une coordination des divers actions et pour détecter la fin de l'activité commune (par exemple lorsqu'un agent détecte que le but a été atteint).

Une fois que l'activité commune est terminée, la coalition disparaît, et le cycle recommence.

Par notre principe de non-bienveillance (P1), nous ne considérons pas que les agents qui reçoivent une proposition de formation de coalition doivent nécessairement l'accepter. Lorsqu'une telle proposition est refusée, l'agent qui l'a envoyée

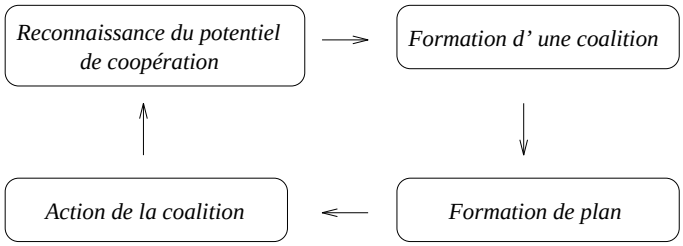


Figure 3.1 : Formation de coalitions

essaie d'en former une autre, avec d'autres partenaires. En ce qui concerne cet aspect, les auteurs de ce modèle n'expliquent pas ni pourquoi un agent à qui une proposition de formation de coalition a été envoyé peut vouloir l'accepter ni pourquoi un agent peut vouloir aussi refuser d'en faire partie [WJ94, page 22].

Nous montrons dans la partie II de ce manuscrit que les relations et situations de dépendance peuvent expliquer ces aspects. D'une part, un agent peut bien refuser de faire partie d'une coalition car il préfère faire partie d'une autre. D'autre part, une relation de dépendance bilatérale peut justifier le fait qu'un agent adopte le but d'un autre, et de cette façon accepte de faire partie d'une coalition.

Dans la suite du manuscrit, nous allons ainsi utiliser le terme *coalition* pour désigner cette notion d'organisation créée et détruite dynamiquement.

3.3 Importance du raisonnement social

Le modèle de formation de coalitions proposé dans la section précédente suppose implicitement que les agents *exploitent une représentation qui leur est interne* concernant certaines propriétés des autres, comme par exemple leur savoir-faire, pour leur permettre de détecter un éventuel potentiel de coopération. Dans le contexte d'un SMA ouvert, cette représentation doit être mise à jour dynamiquement, car des agents peuvent y sortir ou y entrer d'une façon non-prévisible.

Comme nous l'expliquerons plus tard, dans la section 5.1.2, nous n'adoptons pas l'hypothèse que les informations qu'un agent possède sur les autres sont toujours correctes et complètes. Ainsi, nous croyons que la procédure de formation de coalition, plus particulièrement lorsque cette dernière n'est pas réussie, peut servir comme une source d'activation pour la révision de croyances. Par conséquent, il existe une autre raison pour laquelle un agent peut refuser de faire partie d'une coalition : lorsqu'il n'a pas le savoir-faire sur lequel la proposition faite a été basé. De cette façon, l'agent qui a envoyé la proposition doit *réviser sa représentation de l'autre*[2], comme nous l'avons exprimé dans la section 1.2.

En nous rappelant notre définition de raisonnement social présentée dans la

[2]Ici, selon notre principe de la sincérité (P2), nous supposons que les agents ne disent que la vérité à propos de leurs savoir-faire.

section 1.2, il devient clair qu'un mécanisme capable de représenter, raisonner et réviser les informations sur les autres est essentiel pour pouvoir mettre en œuvre une telle procédure de formation de coalitions.

3.4 Modèle d'agent

En considérant la première définition d'agent donnée dans la section 2.3.1, un agent doit être capable de réaliser plusieurs traitements. Dans le contexte de notre étude, un de ces traitements consiste à raisonner sur les autres, et pour cela un agent doit être capable de représenter en interne certaines de leurs propriétés.

Nous appelons *description externe* la représentation qu'un agent a des autres[3] [DM91, PLE92]. Nous ne voulons pas discuter en ce moment quelles sont les propriétés envisageables pour bien permettre de caractériser un agent face aux autres, cette discussion sera faite dans la section 5.1.

Nous nous sommes inspirés du modèle d'agent ASIC [Boi93, BD94b] pour concevoir notre modèle d'agent. Comme notre objectif scientifique n'est que celui de montrer l'intérêt du mécanisme de raisonnement social, nous avons utilisé une version très simplifiée de ce modèle[4]. Notre modèle d'agent est défini de la façon suivante :

Définition 3.4.1. *Soit* $M^{Ag} = \{W, A, IS, perceive, receive, reason, decide, update, send, act\}$ *un modèle pour caractériser le comportement d'un agent, tel que :*

- W *désigne son environnement, c'est-à-dire, les entités passives du monde extérieur avec lesquelles l'agent est en relation ;*
- A *désigne les autres agents faisant partie de la société ;*
- $IS = \{RS, DS, CS\}$ *désigne son état interne, composé par :*

[3] D'autres auteurs appellent cette représentation modèle d'accointances. Initialement, l'expression description externe était associée à un niveau d'abstraction qui "caractérise le système fonctionnellement, essentiellement par rapport à son environnement, à l'aide d'un ensemble d'entrées/sorties" [PLE92, page 3]. Nous utilisons ce terme différemment, car nous croyons que l'expression désigne bien le fait que le niveau de description que les agents utilisent pour représenter les autres n'est pas le même que celui utilisé pour les décrire selon un point de vue interne, c'est-à-dire, ses traitements, ses connaissances etc. Dans un certain sens, c'est bien au niveau de connaissances [New82] que nous situons cette description.

[4] Par rapport au modèle présenté dans [Boi93, page 107], nous avons fusionné les fonctions appelés capteur et interprétation dans *perceive*, exécution et effecteur dans *act*, récepteur et interprétation dialogue dans *receive*, exécution dialogue et émetteur dans *send* et finalement décision et engagement dans *decide*. La fonction *update* n'existait pas dans le modèle, et nous la considérons comme distincte de la fonction *reason*, puisqu'il ne s'agit pas d'inférer des nouvelles croyances. Nous avons aussi simplifié la description des états internes. Par ailleurs, dans les conclusions de ce travail [Boi93, page 228], l'auteur cite comme perspective l'utilisation de son modèle pour exprimer ce qu'il appelle le contrôle social, réalisé par un traitement appelé *organisation*, qui n'a pas été très développé dans le cadre du travail en question. Nous croyons que notre mécanisme de raisonnement social adresse justement cet aspect-là.

  - ⋆ *un état de raisonnement RS, qui contient ses croyances courantes ;*
  - ⋆ *un état de décision DS, qui contient les possibles suites d'actions qui peuvent être choisies dans un moment déterminé ;*
  - ⋆ *un état d'engagement CS, qui précise quel est la suite d'action choisie dans un moment précédent, en établissant des contraintes qui y doivent être appliquées depuis. Il s'agit, par exemple, de préciser que dans un moment un agent est engagé à poursuivre un tel but ou bien qu'il est engagé à exécuter un tel plan ;*

- *perceive* : $W \times A \mapsto RS$ *est une fonction qui permet à un agent de percevoir son environnement et les autres agents*[5] *;*
- *receive* : $A \mapsto RS$ *est une fonction qui permet à un agent de mettre à jour son état de raisonnement à partir des informations qui lui sont communiquées par d'autres agents ;*
- *reason* : $RS \times CS \mapsto DS$ *est une fonction qui permet à un agent d'inférer un ensemble de suite d'actions possibles à partir de de son état de raisonnement et de son état d'engagement courant ;*
- *decide* : $DS \mapsto CS$ *est une fonction qui permet à un agent de décider quelle suite d'actions il doit suivre ;*
- *update* : $CS \mapsto RS$ *est une fonction qui permet à un agent de mettre à jour son état de raisonnement, dans le cas où celui-ci a des croyances inconsistantes ;*
- *send* : $CS \mapsto A$ *est une fonction qui permet à un agent d'envoyer des messages aux autres agents ;*
- *act* : $CS \mapsto W$ *est une fonction qui permet à un agent d'agir sur son environnement.*

En ce qui concerne leur description interne, les agents eux-mêmes peuvent être composés de plusieurs sous-entités responsables pour la mise en œuvre de ces traitements, que pour une question d'uniformisation sémantique nous préférons appeler *mécanismes internes* plutôt que sous-agents comme dans [BRCHV95] ou encore agents locaux comme dans [CC93b]. Un mécanisme de contrôle doit assurer que chacun de ces mécanismes internes soient activés dans le bon ordre[6].

[5] Nous voulons dire par cette affirmation qu'un agent "intelligent" doit être capable de percevoir certaines propriétés concernant des autres agents.

[6] Au niveau du modèle, nous ne fixons a priori des contraintes ni sur le contrôle de ces mécanismes ni sur leur concurrence : nous pouvons avoir un contrôle centralisé ou distribué, et les mécanismes eux-mêmes peuvent être concurrents ou non. Un exemple d'une approche concurrente est présenté dans [OD94], où la réalisation d'un modèle d'agent est proposé par l'utilisation d'un tableau noir parallèle.

Les mécanismes internes que nous proposons pour la mise en œuvre des traitements décrits ci-dessus sont présentés dans la figure 3.2. En considérant notre modèle, il existe une correspondance totale entre les fonctions définies ci-dessus et les mécanismes internes qui les mettent en œuvre.

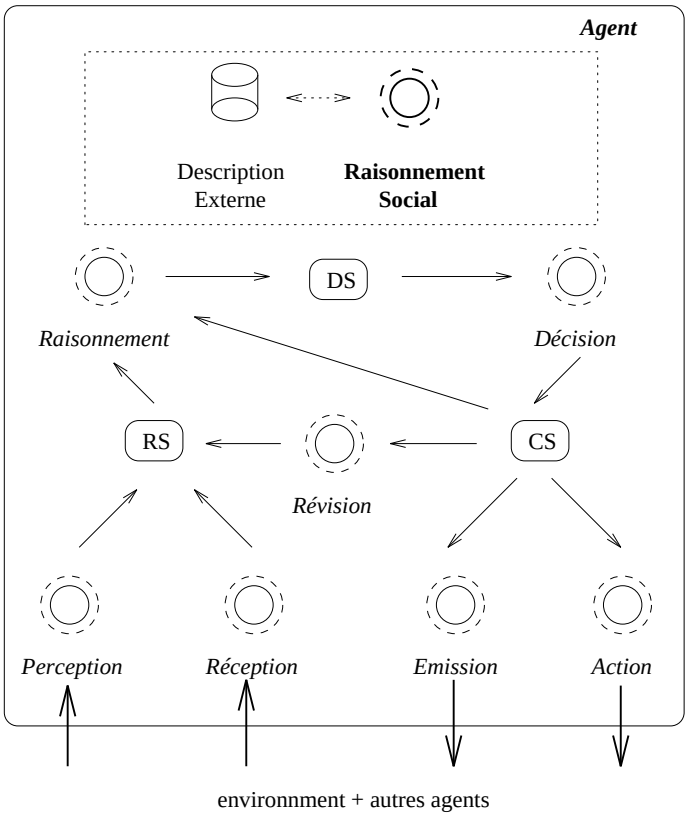


Figure 3.2 : Modèle d'agent

En haut de la figure 3.2, nous représentons en gras un mécanisme qui à notre avis a une signification particulière dans le contexte d'un SMA. Par cela nous voulons dire que son existence — et son exploitation par les mécanismes de raisonnement, de décision et de révision — est due au fait que l'agent est plongé dans un environnement où d'autres agents existent. Il s'agit du *mécanisme de raisonnement social*. Comme nous venons de montrer dans la section 3.3, un agent placé dans un SMA ouvert, où les agents qui en font partie et donc leurs capacités ne sont pas connus a priori, doit être capable de raisonner sur des autres, en particulier pour pouvoir former des organisations de façon dynamique. Ce raisonnement est fait à partir de la *description externe* que les agents ont les uns des autres, cette dernière pouvant être considérée comme une instance particulière de l'état de raisonnement RS[7]. En ce qui concerne la révision de croyances, comme

[7]Néanmoins, comme nous l'avons dit précédemment, la description externe est représentée

les agents n'ont qu'une vision partielle de leur environnement et des capacités des autres agents, un agent doit pouvoir changer ses croyances au fur et à mesure qu'il interagit avec les autres. Nous nous intéressons particulièrement dans ce travail aux croyances que les agents ont les uns sur les autres, c'est-à-dire, à la révision de leurs descriptions externes. Nous explicitons ce point en plus de détail dans le chapitre 8.

Dans notre contexte d'étude, nous considérons qu'*il n'existe pas de planification en ligne* : soit un agent a des plans pour atteindre un certain but, soit il va adopter le plan proposé par un autre lorsqu'il accepte participer d'une coalition. Néanmoins, nous indiquons dans la section 6.3.3 comment cette restriction peut être enlevée.

Dans cette thèse, nous ne sommes intéressés que par le *mécanisme de raisonnement social*. Nous allons détailler les effets d'un tel mécanisme dans la partie II de ce manuscrit.

3.5 Modèle de société

Dans la figure 3.3 suivante, nous présentons notre modèle d'une société d'agents, défini de la façon suivante :

Définition 3.5.1. *Soit $M^{Soc} = \{A, L\}$ un modèle de société, tel que :*

- *A désigne les agents qui en font partie ;*
- *L désigne un langage d'interaction qui garantit que les agents puissent comprendre les messages qu'ils envoient les uns aux autres.*

Une *société* est un ensemble d'agents pouvant potentiellement interagir. Pour sa part, chaque *agent* est composé d'un ensemble de *mécanismes internes*, dont son comportement global est dicté par un *contrôle* local.

En ce qui concerne l'ouverture de la société, la seule contrainte que nous imposons est que les agents doivent communiquer aux autres leur arrivée et leur sortie. En arrivant, un agent doit se présenter aux autres, en les envoyant les informations sur soi-même, afin de leur permettre de mettre à jour leurs descriptions externes. Lorsqu'un agent sort de la société, il envoie aussi un message aux autres, qui enlèvent de leurs descriptions externes l'information concernant l'agent partant[8].

Au niveau de la société, nous n'imposons a priori aucune contrainte sur les mécanismes internes d'un agent. Nous voulons mettre en relief le fait que nous

dans un niveau d'abstraction plus haute. Si nous voulions être précis, il fallait d'abord traduire son contenu dans le langage interne de l'agent. Pour une question de concision, nous considérons que cette traduction est déjà faite.

[8]Si nous supposons que les agents ont un mécanisme de perception assez élaboré, nous pourrions enlever ces restrictions.

ne supposons pas que les agents soient homogènes : des agents peuvent avoir des mécanismes internes différents, même si, dans le cadre de ce travail, selon le modèle d'agent que nous venons de proposer, nous nous intéressons à un mécanisme de raisonnement social, commun à tous les agents qui font partie de la société, représenté en gris dans la figure 3.3.

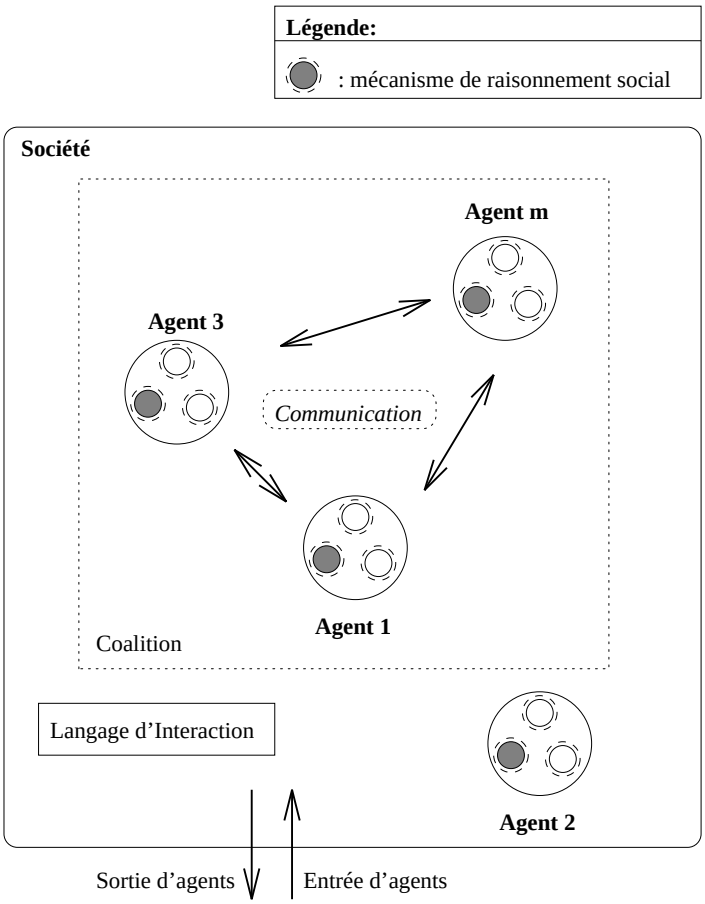


Figure 3.3 : Modèle de société

3.6 Synthèse

Dans ce chapitre, nous avons montré que dans le contexte des systèmes auxquels nous nous intéressons, c'est-à-dire un SMA ouvert où les agents ne sont pas nécessairement bienveillants, une approche dynamique et décentralisée pour la conception de l'organisation de la société d'agents est envisageable.

Nous avons adopté une méthode pour la formation de coalition entre agents, et nous avons proposé un modèle d'agent et un modèle de société qui permettent la

mise en œuvre de cette méthode. Particulièrement, nous avons appelé *mécanisme de raisonnement social* le mécanisme interne d'un agent responsable pour cette mise en œuvre. Nous ne supposons pas que les agents soient tous homogènes en ce qui concerne tous leurs mécanismes internes, la seule restriction que nous imposons est qu'ils le sont en ce qui concerne le mécanisme de raisonnement social.

Dans cette thèse, nous ne nous intéressons qu'à la formation de coalitions. En rappelant la méthode de résolution coopérative de problèmes présentée dans la section 3.2, nous n'abordons ni l'étape de formation de plans ni celle relative à l'action de la coalition. Nous considérons plutôt que les agents ont un ensemble de plans pré-établis pour atteindre leurs buts. Dans la partie II suivante, nous montrons comment un mécanisme de raisonnement social, fondé sur la notion de dépendance entre agents, peut être utilisé pour mettre en œuvre la formation dynamique de coalitions entre agents.

Deuxième partie

Mécanisme de raisonnement social

Chapitre 4

Théorie de la dépendance

Dans ce chapitre, nous présentons les éléments principaux de la théorie de la dépendance dont nous nous sommes inspirés pour concevoir notre mécanisme de raisonnement social. Selon cette théorie, la cause de certains types d'interaction sociale comme la coopération et l'échange social se situe dans la *complémentarité* des agents. Nous montrons que cette théorie fournit une base à partir de laquelle nous pouvons expliquer et prédire les interactions sociales que les agents peuvent avoir. Les deux notions principales de cette théorie sont la *dépendance* et le *pouvoir*. Quand un agent sait qu'il dépend d'un autre agent pour atteindre un de ses propres buts, il va essayer d'influencer ce dernier pour adopter ce but. Plus un agent a le pouvoir d'influencer l'autre (par exemple en exploitant le fait que l'autre dépend aussi de lui), plus la possibilité que l'autre agent accepte d'adopter son propre but est grande. Néanmoins, ces relations de dépendance et de pouvoir sont *objectives* : elles existent même si les agents n'en sont pas conscients. Dans le cadre de cette thèse, nous utilisons le mot *subjective* pour désigner une représentation mentale interne aux agents, mais nous ne considérons pas ces deux notions (représentation objective et subjective) nécessairement en opposition. Ce chapitre reprend quelques idées que nous avons présentées dans [CS95].

Ce chapitre est organisé de la façon suivante. Afin de situer cette théorie dans le contexte plus large des travaux relatifs aux interactions sociales, nous présentons dans la section 4.1 les différentes classes de modèles possibles utilisés pour les caractériser, et nous comparons ces modèles entre eux. Le terme d'autonomie est souvent utilisé pour caractériser le comportement d'un agent. La section 4.2 présente les différents significations qui lui sont associées. Les concepts d'interférence, d'action et d'interaction sociale sont décrits dans la section 4.3. La section 4.4 explicite les différentes raisons pour lesquelles un agent autonome peut vouloir adopter les buts d'autrui. Nous présentons ensuite dans la section 4.5 les éléments de base de la théorie de la dépendance, et dans la section 4.6 certains types de relations de dépendance particulières. Dans la section 4.7, nous expliquons comment à partir de la représentation *subjective* de ses dépendances (c'est-à-dire appartenant à son état mental) un agent peut générer des buts so-

ciaux, c'est-à-dire, décider d'entreprendre une certaine action sociale et comment les interactions sociales ont lieu. Enfin, nous terminons ce chapitre en présentant dans la section 4.8 une discussion sur ce modèle et sur son utilisation dans notre contexte de recherche.

4.1 Modèles d'interaction sociale

Dans [CC92], deux classes différentes de modèles pour l'interaction sociale sont présentées :

- les modèles *descendants*("top-down") : selon ces modèles, on considère que les agents ont a priori un problème global à résoudre. La coopération est pré-établie comme hypothèse de départ. Les interactions sociales sont alors contraintes par une structure organisationelle conçue de façon statique, telle que nous l'avons présentée dans la section 3.1. Celle-ci guide les agents pour atteindre le but global pour lequel le système a été conçu. Ces modèles sont utilisés dans la plupart des SMA qui adoptent une perspective RP ;

- les modèles *ascendants* ("bottom-up") : selon ces modèles, les agents n'ont pas de buts communs a priori. Les interactions sociales qui ont lieu sont le résultat de leurs tentatives pour atteindre leurs propres buts. Les structures organisationelles, ainsi que la coopération, ne sont pas pré-établis comme hypothèse de départ. Ces modèles sont utilisés plutôt dans les SMA qui adoptent une perspective SS.

En ce qui concerne les modèles ascendants, nous avons montré dans [CS95] que l'on peut les décomposer en deux sous-classes :

- les modèles fondés sur l'*utilité* : selon ces modèles, dont la théorie des jeux est un exemple [LR57, Axe84], le monde social repose sur le principe "*bellum omnium contra omnes*" : il est considéré comme un domaine d'interférences entre agents. Ceux-ci doivent nécessairement se coordonner entre eux, en créant des conventions et des contraintes, afin d'avoir un comportement global cohérent. L'existence des autres agents *limite* ainsi l'autonomie et le pouvoir de chaque agent appartenant à la société ;

- les modèles fondés sur la *complémentarité* : ces modèles, sur lesquels nos travaux sont fondés, proposent une vision différente de l'interaction sociale. Ils prennent en compte l'existence possible de capacités complémentaires entre les agents, par rapport aux buts à atteindre. Ainsi, l'existence des autres agents *augmente* l'autonomie et le pouvoir de chaque agent, car même si un agent ne peut pas atteindre tout seul un de ses propres buts, il peut néanmoins l'atteindre en demandant de l'aide aux autres.

Une synthèse de ces approches est présentée dans la figure 4.1.[1].

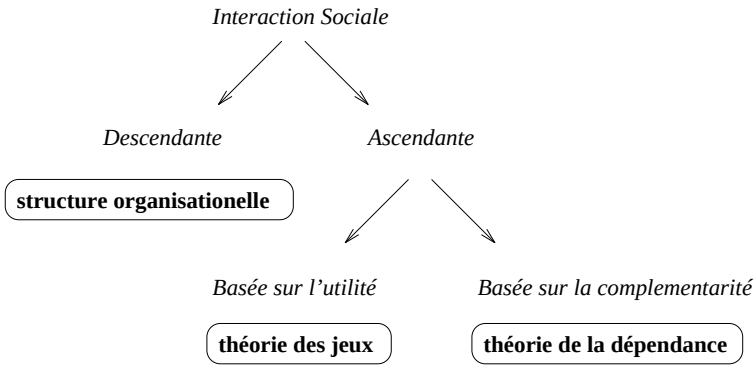


Figure 4.1 : Modèles d'interaction sociale

4.1.1 Modèles fondés sur les structures organisationelles

Normalement, dans une perspective RP, l'interaction sociale sert à *coordonner* un ensemble d'agents, qui ont déjà un but commun à accomplir. L'allocation de tâches entre des agents peut être pré-établie par le concepteur ou bien établie dynamiquement par des contrats [Smi80]. Néanmoins, même avec une allocation dynamique, les tâches elles-mêmes et les différents rôles de résolution sont définis à priori, le problème étant d'attribuer les bons rôles aux bons agents [Mal87]. Les relations objectives, telles que nous les considérons, ne sont pas représentées au sein de ces systèmes. Parfois, elles existent en tant que structures au niveau de l'implémentation. Nous pouvons par exemple citer les mécanismes de synchronisation basés sur une architecture de tableau noir [EHRLR80, EM88].

Nous croyons qu'une approche fondée exclusivement sur les relations organisationelles pour modéliser l'interaction sociale a les inconvénients suivants [CC92] :

- *absence d'une perspective dynamique* : les agents sont tellement au courant des conditions de leurs interactions que le processus d'émergence des interactions à partir des actions sociales est complètement ignoré ;

- *subjectivisme social* : les relations sociales sont étudiées seulement à partir des états mentaux des agents. On commence l'analyse en supposant que des buts et des croyances communs existent déjà et sont représentés dans leurs états mentaux ;

[1] Il faut signaler que nous avons représenté dans cette figure les deux types de modèles ascendants comme étant complètement disjointes pour une question didactique. A vrai dire, nous croyons que la notion de complémentarité recouvre la notion d'utilité, en y ajoutant explicitement la notion de but. Nous revenons à cette comparaison dans la section 4.1.2.

- *mise en relief de la communication* : comme conséquence des deux points ci-dessus, l'action sociale n'est conçue que par communication, dont l'objectif est de modifier les buts et les croyances communes [Gal88, Pol90]. Comme on montre dans [Cas92], la communication n'est qu'un moyen d'action sociale.

Une autre point à considérer est que les structures organisationelles rigides ne permettent pas de prendre en compte l'*évolution des interactions* au sein d'une société ouverte. Dans une société ouverte, au fur et à mesure de l'entrée/sortie d'agents de la société, de nouvelles capacités sont rendues disponibles/indisponibles. Ceci transparaît dans l'interférence sociale entre agents :

- lors de l'entrée de nouveaux agents, des buts qui ne pouvaient être poursuivis par manque d'agents capables d'entreprendre certaines actions peuvent être maintenant poursuivis, si les agents qui arrivent ont la capacité d'entreprendre ces actions ;

- en ce qui concerne le pouvoir social des agents, une réorganisation de leur réseaux de dépendances peut l'augmenter/diminuer, lors de l'arrivée de nouveaux agents.

Comme exemple de cette approche, nous pouvons citer les nombreux travaux développés dans le cadre du projet ARCHON [OCR91, JW92, JVA+93, JP93]. Un modèle pour l'adaptation des structures organisationnelles dans un SMA est présenté dans [LS95]. Selon la tâche à accomplir, les agents sont capables de choisir la forme d'organisation la plus appropriée parmi un ensemble de base.

4.1.2 Modèles fondés sur l'utilité

Un des paradigmes les plus utilisés pour caractériser la rationalité est l'utilitarisme. Selon ce paradigme, un agent autonome est dit rationnel s'il cherche toujours à maximiser sa propre utilité espérée [Doy92]. Cette notion de rationalité est la base de nombreuses théories modernes d'économie. Les éléments de base pour définir l'utilitarisme sont la théorie de l'utilité et la théorie de la décision. La théorie de l'utilité propose un modèle pour caractériser les préférences d'un agent entre les divers états possibles du monde, en reflétant ses désirs de haut niveau. La théorie de la décision étend cette dernière en permettant de représenter des opérations quantitatives sur des préférences (comme l'addition) et en rendant possible le calcul des préférences lorsque de l'incertitude est présente (utilité espérée). La théorie des jeux est construite à partir de ces deux théories, où on s'intéresse à une source d'incertitude très particulière : celle qui prend en compte l'action d'autres agents également rationnels [GD93].

Dans le cadre de cette théorie, un jeu est généralement caractérisé par une matrice, où sont représentés les gains (utilités) des joueurs qui choisissent des actions représentées dans les lignes et les colonnes. Un problème classique traité

par la théorie des jeux est celui du *dilemme du prisonnier*[2] , dont la matrice de gain est présentée dans la figure 4.2 [Axe84].

		K	
		c	*d*
J	*c*	3 3	5 0
	d	0 5	1 1

Figure 4.2 : Dilemme du prisonnier [Axe84]

Nous croyons qu'un approche fondée exclusivement sur l'utilité pour modéliser l'interaction sociale a les inconvénients suivants :

- normalement, les agents sont considérés comme *homogènes* et *auto-suffisants*. Autrement dit, ils peuvent toujours atteindre leurs buts tout seuls. Les interactions sociales ont lieu soit pour maximiser l'utilité de chacun soit pour minimiser l'interférence sociale. Dans ce deuxième cas, les agents arrivent parfois à un compromis par rapport à ses buts initiaux, comme dans [RZ94] ;

- en ce qui concerne l'émergence de la coopération, les résultats obtenus jusqu'à présent par la théorie des jeux ne sont pas très satisfaisants, car on suppose que les agents ne se trompent jamais dans leurs choix, qu'ils ont une connaissance correcte et complète les uns des autres et que cette connaissance est statique, c'est-à-dire, leurs stratégies n'évoluent pas au fur et à mesure que les agents interagissent entre eux [Lom93] ;

- la notion de *but* n'est jamais représentée explicitement. Pour cette raison, la théorie des jeux n'arrive pas à distinguer les divers formes d'action et d'interaction sociales telles que nous les présentons dans la section 4.3 ;

- dans le cadre de la théorie des jeux, l'action sociale est purement *stratégique*. Autrement dit, un agent ne prend en compte que les effets possibles des actions des autres pour prendre ses décisions. Les agents sont considérés comme des entités passives, incapables de *changer et d'influencer* les actions

[2]Supposons que la police ait mis en garde à vue deux prisonniers, dans deux cellules différentes de façon à les empêcher de communiquer. L'inspecteur propose aux deux prisonniers de dénoncer l'autre comme auteur du crime (action *d*). Celui qui dénonce aura une réduction de peine. Clairement, pour chaque prisonnier, la meilleure action est de dénoncer en espérant que l'autre ne le dénonce pas. Si nous appliquons une technique appelée analyse de dominance, les deux se dénoncerons l'un à l'autre et la réduction de peine pour chacun (1) sera plus petite que celle dans le cas où ils coopèrent (action *c*, avec une réduction de 3).

possibles des autres. Comme nous le montrons dans la section 4.7, c'est justement à partir des actions sociales d'influence et d'adoption qu'on peut expliquer la raison pour laquelle une coopération a lieu.

Dans le cadre strict de la théorie des jeux, les agents ont une connaissance a priori des préférences des autres agents, et par conséquent la communication n'a pas lieu. L'effet de bord de cette hypothèse est que les agents sont *omniscients*. A notre connaissance, les premières tentatives pour combiner la communication avec le modèle d'action rationnelle proposé par cette théorie ont été présentées dans [RG85]. Par la suite, une grande partie des travaux dans le domaine des SMA se sont inspirés de ce modèle de rationalité, nous pouvons citer [ZR90, ZR93, ER93, Kra93, GH94]. Une référence de base est sans doute [RZ94].

4.1.3 Modèles fondés sur la complémentarité

Selon le modèle utilisé dans le cadre de ce travail, l'interaction sociale est fondée sur des modèles de relations structurelles *objectives* dans lesquelles les agents sont plongés. Une de ces relations, que nous considérons comme l'une des plus fondamentales et sur laquelle nous allons nous concentrer est celle de la *dépendance sociale* : les agents ont souvent besoin l'un de l'autre pour accomplir leurs buts. En devenant *subjective* (appartenant à l'état mental des agents), la dépendance peut expliquer à la fois les raisons pour lesquelles les agents adoptent les buts les uns des autres et celles pour lesquelles certaines interactions sociales telles que la coopération et l'échange social apparaissent au sein d'une société. Nous détaillons ces idées par la suite.

4.2 Autonomie d'un agent

Comme dans toute discipline récente, il n'existe pas encore au sein de la communauté SMA de vocabulaire commun qui associe à chaque terme utilisé dans le domaine un sens précis qui soit à la fois unique et globalement reconnu. C'est le cas en ce qui concerne le terme agent, comme nous l'avons montré dans la section 2.3.1. Et même des tentatives visant à définir des taxonomies de termes sont actuellement développées [PLE92].

En particulier, nous trouvons souvent dans la littérature des SMA (et plus généralement dans celle de l'IA) le terme *autonomie* associé à au moins quatre significations très différentes :

- *autonomie par rapport à la conception* : cette signification, utilisée dans la section 2.2.1, fait référence à la différenciation que nous avons proposé entre les démarches de conception des systèmes de RDP et des SMA selon le critère de réutilisation. Dans ce contexte, un agent est autonome s'il a une existence propre, indépendamment de l'existence des autres agents [DM90b] ;

- *autonomie par rapport à l'environnement* : un agent autonome est un agent censé fonctionner dans des environnements dynamiques et incertains, qui ne peuvent être perçus que de façon imparfaite, qui peuvent changer suite à des actions qui ne sont pas contrôlées par l'agent lui-même et sur lesquels les effets de ses actions ne sont pas toujours prévisibles. Cette définition d'autonomie est typiquement rencontrée dans les travaux en robotique mobile [Nil94]. Elle est assez proche aussi de la notion de autopoïèse [Bou95] ;

- *autonomie par rapport à ses propres buts* : un agent autonome est un agent qui peut atteindre ses buts tout seul. Il n'a pas besoin a priori de coopérer avec d'autres agents, et s'il décide de le faire, c'est à cause d'une amélioration possible de performance. Cette deuxième notion d'autonomie, de même qu'une autre concernant la localité (locale ou globale) de la tâche à exécuter, a été utilisée dans [DM90b] pour classer les comportements possibles d'un agent (cohabitation, coopération, collaboration et distribution) au sein d'une société ;

- *autonomie par rapport à ses motivations* : un agent autonome est un agent qui a le libre choix pour interagir socialement. C'est à partir du contenu de son état mental qu'il décide de coopérer ou non, d'adopter les buts d'autres agents ou non etc. [Cas90]. Autrement dit, la signification du terme ici recouvre le principe de la non-bienveillance (P1) énoncé dans la section 1.3.

A notre avis, ces définitions ne sont pas comparables, car elles adressent des aspects très différents. Nous ne voulons pas ici proposer une autre non plus, car à notre avis cela n'apporte rien au domaine. Dans le cadre de notre travail, nous les utilisons toutes, dépendant de l'aspect que nous voulons aborder : (i) lorsque nous parlons de réutilisabilité du logiciel, comme discuté dans la section 2.2.3, nous utilisons la première signification, (ii) lorsque nous parlons d'adaptation d'un agent en ce qui concerne l'ouverture du système, nous prenons en compte la deuxième signification, (iii) lorsque nous présentons notre modèle formel pour la théorie de la dépendance dans le chapitre 5, nous associons au terme autonomie la troisième signification et (iv) tout au long de ce chapitre, nous adoptons la quatrième signification.

Nous utilisons ainsi les propriétés proposées dans [Cas90] pour caractériser un agent autonome selon ce dernier sens présenté ci-dessus :

1. un agent autonome a ses propres buts ;

2. il est capable de prendre des décisions concernant ses propres buts (conflits internes) ;

3. il est capable de décider de façon autonome quand il doit adopter les buts des autres agents. L'adoption de buts n'est ni automatique, ni gouvernée par des lois simplificatrices, telles que adopter toujours les buts des autres agents sauf si ces derniers sont en conflit avec les siens ;

4. il considère l'action sociale d'adopter les buts des autres comme un moyen d'atteindre ses propres buts ;

5. il contrôle l'acquisition de ses croyances et leur crédibilité par rapport à ses croyances courantes.

Ces propriétés peuvent être résumées par les deux postulats suivants [Cas90] :

- au sein d'un agent, chaque but est généré à partir des buts (croyances) pré-existants (par instanciation, planification ou "means-end analysis") ;
- chaque modification des buts des autres agents n'est réalisée que par la modification de leurs croyances.

Un des objectifs de nos travaux est de montrer qu'une telle définition d'autonomie peut aussi être utilisée dans une perspective RP, en particulier en ce qui concerne la formation de coalitions. Nous illustrons ce point dans le chapitre 11.

4.3 Interférence, action et interaction sociales

Dans certains courants sociologiques, les buts ne sont pas pris en compte comme des objets d'étude relevants dans la mesure où ils ne sont pas parfaitement définis [CS95]. Par exemple, dans les théories de l'action rationnelle, ils sont masqués par l'utilisation de la notion d'utilité, considérée comme le mécanisme fondamental qui caractérise les systèmes autonomes. Cependant, nous ne croyons pas que la motivation principale des agents pour interagir soit le principe de maximisation de l'utilité. Les actions sont motivées par les désirs, les besoins, etc. Autrement dit, *les actions sont entreprises pour atteindre des buts.* De manière analogue, dans la société humaine, c'est la volonté d'accomplir un but pré-existant qui pousse des personn8s à coopérer. Par conséquent, à notre avis, la question principale à poser pour modéliser l'interaction sociale n'est pas pourquoi des agents choisissent ou non de coopérer, mais pourquoi finalement la coopération en tant que telle est en fait prise en compte.

Pour répondre à cette question, nous devons d'abord introduire trois notions de base : l'interférence sociale, les actions sociales et l'interaction sociale [CC92].

4.3.1 Interférence sociale

Nous considérons qu'il existe une *interférence sociale* entre deux agents i et j lorsque le fait que i atteigne un de ses buts a un effet sur un ou plusieurs buts de j. Deux types d'interférence existent :

- interférence *positive* : le fait que i atteigne un but g produit un état du monde qui sert les objectifs de j, c'est-à-dire, plus proche d'un de ses buts g' ;

- interférence *négative* : le fait que i atteigne un but g produit un état du monde qui ne sert pas et même contrarie les objectifs de j, c'est-à-dire, un de ses buts g' ne peut plus être atteint. Nous pouvons caractériser deux types d'interférences négatives :

  - ⋆ *conflit* : les buts g et g' sont en conflit s'il n'existe pas un état du monde où g et g' peuvent être atteints en même temps ;
  - ⋆ *concurrence* : dans le cas où les ressources sont limitées, les buts g et g' sont concurrents s'ils ne peuvent pas être atteints par les deux agents au même instant, même s'ils ne sont pas en conflit au sens précédent.

Cette notion d'interférence est assez proche de celle définie dans [vM90b, vM90a] dans le cadre de la planification multi-agents, mais nous considérons que l'interférence est une notion *objective*, qui existe indépendamment de sa représentation dans l'état mental des agents.

4.3.2 Action sociale

Nous désignons par *action sociale* une action d'un agent qui prend en compte dans la séquence de buts qui la génère une des alternatives suivantes : (i) les actions pouvant être entreprises par d'autres agents ou (ii) les attitudes mentales (buts, croyances, émotions) de ces derniers [CC92].

Pour mieux caractériser les actions sociales, supposons deux agents i et j où g est un but de i. Nous pouvons distinguer les actions sociales suivantes de la part de i :

- *exploitation* : si j a aussi le but g et si j peut atteindre ce but tout seul, i n'a qu'à attendre que j le fasse ;
- *influence* : si i ne peut pas atteindre son but, et s'il croit que j peut l'aider à atteindre son but d'une certaine manière (interférence positive), i peut essayer d'influencer j pour adopter le but g, c'est-à-dire, le but de faire que j ait lui aussi le but g[3] devient un des buts de i. La capacité de i à convaincre j d'adopter son but sera plus ou moins efficace en fonction de son pouvoir d'offrir à j quelque chose en échange, comme nous le verrons dans la suite ;
- *agression* : si j a un autre but g' qui est en conflit avec g, i peut essayer d'empêcher j d'atteindre g' ;
- *adoption* : si pour une certaine raison j désire que i puisse atteindre son but g, il adopte ce but, c'est-à-dire que g devient un but de j pour la simple raison que g est un but de i. Nous allons analyser en détail dans la section 4.4 les différentes raisons pour lesquelles un agent peut vouloir adopter un but d'un autre agent.

[3] Ou bien un des sous-buts qui en découlent.

Comme les actions des agents se produisent à partir de ses connaissances, nous avons implicitement adopté l'hypothèse que les interférences sociales décrites ci-dessus sont devenues *subjectives* et explicitement représentées dans l'état mental des agents.

4.3.3 Interaction sociale

Nous désignons par *interaction sociale* une situation observable au niveau de la société d'agents qui est la conséquence des actions sociales des agents. Nous aurons ainsi différentes formes d'interaction sociale selon les combinaisons possibles d'actions sociales entreprises par chacun des agents. De façon similaire aux interférences, nous pouvons aussi caractériser deux types d'interaction sociale :

- interactions sociales *positives* : les agents s'entraident. Deux exemples de ces interactions nous intéressent particulièrement :

  - ⋆ *coopération* : supposons que (i) i et j ont le même but g (ii) aucun de deux ne peut atteindre ce but tout seul et (iii) ensemble, ils peuvent atteindre g. Par exemple, supposons que i et j soient *complémentaires* par rapport à g : chacun peut entreprendre une action qui permet que g soit atteint. Dans cette situation, i peut proposer à j une coopération pour atteindre g (influence) et j peut accepter la proposition de i (adoption) ;
  - ⋆ *échange social* : supposons maintenant que j n'ait pas le but g, mais un autre but g' que i peut l'aider à atteindre. Dans ce cas, i peut proposer à j de l'aider à atteindre son but g' si en contrepartie ce dernier l'aide à atteindre g (influence) et j peut accepter de nouveau la proposition de i (adoption). Nous allons montrer dans le chapitre 7 que ce type d'interaction est moins intéressant pour un agent que la coopération, car la notion de réciprocité doit être prise en compte ;

- interactions sociales *négatives* : les agents essaient de s'empêcher mutuellement d'atteindre leurs buts en conflit ou en concurrence. Comme un exemple de ce type d'interaction, nous avons :

  - ⋆ *compétition* : si i et j sont *homogènes* (ils ont les mêmes buts et sont capables d'entreprendre les mêmes actions dans le monde) et leurs buts sont concurrents, chacun d'entre eux peut essayer d'empêcher l'autre d'atteindre le but commun.

Dans la suite de ce travail, nous nous limitons à l'étude des actions sociales d'influence et d'adoption, car nous croyons qu'elles sont les plus significatives pour notre contexte d'étude. Par conséquent, nous nous limitons aussi aux interactions sociales positives de coopération et d'échange social.

4.4 Adoption de buts

Selon [Cas92], il existe une contradiction entre le but de modéliser des agents rationnels et de leur associer par défaut un comportement de *bienveillance*, c'est-à-dire qu'ils adoptent inconditionnellement les buts les uns des autres. Un modèle d'agent rationnel doit considérer qu'un coût est associé aux actions, et que les ressources sont limitées. Un agent ne peut pas être considéré comme rationnel s'il entreprend des actions ou utilise ses ressources pour permettre aux autres d'atteindre leurs buts. En agissant de cette sorte, son comportement est autant irrationnel que dans le cas où il essaye d'atteindre de buts irréalisables : dans les deux cas, son gain est nul.

Nous devons tout d'abord remarquer que l'adoption de buts est un cas particulier d'une problématique plus générale : celle de la genèse des buts. Cette problématique a été fortement négligée jusqu'à présent dans la littérature des SMA. Dans la plupart des approches, formelles [CL87, Wai94b, RGS92], ou applicatives [Lev90, OCR91], l'hypothèse de départ est qu'ils existent déjà.

Afin de mieux expliquer les raisons pour lesquelles un agent décide d'adopter les buts des autres agents, nous croyons utile de distinguer deux grandes classes d'adoption de buts [Cas92] :

- *adoption terminale* : l'adoption de but est en quelque sorte un but de haut niveau en elle-même. Comme dans [CM93], nous considérons que cette classe d'adoption peut être divisée en deux sous-classes :
  - ⋆ *adoption personnelle* : un agent adopte toujours les buts d'un agent donné. L'adoption est indépendante de la nature du but. Elle n'est justifiée que par des relations de nature émotive entre des agents comme l'amour et l'amitié ;
  - ⋆ *adoption non-personn8lle* : un agent adopte certains buts d'un ensemble d'agents bien déterminés. À la différence du cas précédent, la nature du but est prise en compte pour justifier l'adoption. Nous pouvons envisager aussi deux sous-classes pour ce type d'adoption :
    - ◇ *adoption fonctionnelle* : un agent adopte certains buts d'un autre agent à cause d'une structure fonctionnelle dans laquelle il est inséré ;
    - ◇ *adoption normative* : ce genre d'adoption caractérise la situation où n'existent ni une structure fonctionnelle ni une adoption personnelle. L'adoption est justifiée par les normes générales qui régissent une société ;
- *adoption instrumentale* : l'agent adopte le but d'un autre car il profite de ce fait. Cette adoption lui permet/facilite d'atteindre un de ses propres buts. Il faut noter que dans cette classe, ni l'agent qui est l'objet de l'adoption ni

le but qui est adopté ont de l'importance, l'adoption est considérée comme un moyen pour l'agent atteindre ses propres buts.

En ne considérant que la problématique d'adoption de buts, l'*hypothèse de bienveillance* peut être caractérisée comme un cas extrême de l'adoption personnelle, lorsqu'un agent veut satisfaire les buts de n'importe quel autre agent[4].

La plupart des SMA conçus dans une perspective RP, il existe une coopération pré-conçue dans le système, caractérisée par un but global et commun à tous les agents. Ce but est soit explicitement représenté au sein du système final, soit implicitement pris en compte pendant l'activité de conception. Comme nous l'avons analysé précédemment, la mise en œuvre de cette coopération est guidée par une organisation de la société d'agents, qui peut également être explicitement représentée au sein des agents (dans le cas des SMA) ou non (dans la plupart des systèmes de RDP), et donc l'adoption est souvent terminale, non-personnelle et fonctionnelle[5].

En ce qui concerne l'adoption personnelle, nous ne connaissons pas jusqu'à présent de théorie formelle qui l'explique de façon satisfaisante. La raison, à notre avis, est plus ou moins évidente : pour expliquer ce type d'adoption, il faut au préalable une théorie sur les émotions, ce qui est loin d'être une tâche facile. Les émotions sont rarement représentées et manipulées dans les SMA, par contre elles sont très présentes dans la société humaine. Une approche très originale et intéressante sur la modélisation des émotions chez les agents peut être consultée dans la littérature du projet OZ [RB92, BLR93, Bat94].

L'adoption normative est aussi peu exploitée dans le domaine des SMA, car il faut proposer d'abord un modèle de normes. Par exemple, il faut expliquer comment ces dernières sont représentées dans l'état mental d'un agent, comment un agent différencie entre ses buts normatifs et ses buts non-normatifs, comment et quand un agent décide (si possible) de ne pas respecter les normes etc. Quelques travaux concernant la représentation mentale des normes peuvent être trouvés dans [CC93a, CC94, ST92].

Dans la suite de ce travail, nous nous intéressons seulement à l'adoption instrumentale. En particulier, nous voulons démontrer que cette classe d'adoption, étant plus générale que l'adoption terminale, peut être utilisée soit dans une perspective SS, soit dans une perspective RP.

[4] Dans quelques modèles, cette adoption a une contrainte : le but à adopter doit être compatible avec les buts propres de l'agent.

[5] Prenons comme exemple le modèle d'agent ASIC [Boi93, BD94b]. Dans ce modèle, l'organisation du système contraint les interactions possibles entre les agents, en explicitant pour chaque agent quels sont les agents dont il peut recevoir des ordres pour adopter un but ou un plan. Cette organisation est explicitement représentée au sein des agents du système. Selon les classes d'adoption présentées ci-dessus, nous pouvons caractériser les agents de ce système comme ayant une procédure d'adoption de buts terminale, non-personnelle, et fonctionnelle, car les contraintes sont fixées sur les rôles de résolution et non directement sur les agents qui jouent ces rôles. Ainsi, dans ce système, un agent n'est pas bienveillant par rapport à tous les autres agents du système.

4.5 Pouvoir social et relations de dépendance

Dans [Cas90, CMC91, CMC92, Cas92, Con92, CC92, Cas95], une théorie de l'interaction entre agents fondée sur des travaux en psychologie sociale est présentée. Son but scientifique est d'expliquer comment les interactions sociales peuvent être dérivées et prédites à partir de conditions structurelles externes. En particulier, cette théorie propose un modèle pour la formation de *buts sociaux* dans les états mentaux des agents. Dans ce contexte, un but social d'un agent est un but d'entreprendre une certaine action sociale, telle que nous les avons énumérées dans la section 4.3.2.

Pour exprimer les états mentaux des agents, les auteurs proposent un modèle formel de leur théorie en utilisant l'apparatus technique proposé par Cohen et Levesque dans leurs travaux sur les intentions [CL87]. Nous ne détaillons pas ici cette formalisation, pour deux raisons : (i) le but de ce chapitre est de présenter de façon informelle les bases de cette théorie et (ii) nous proposons une formalisation alternative de cette même théorie dans le chapitre 5[6].

Cette théorie fournit la base pour répondre à deux questions fondamentales concernant une société d'agents autonomes [Cas90] :

1. *Le problème de sociabilité* : Pourquoi un agent autonome décide-t-il d'interagir socialement ?

2. *Le problème d'adoption* : Comment un problème appartenant à un agent devienne-t-il un problème social, c'est-à-dire, comment fait un agent pour que les autres agents adoptent ses buts ?

Selon la théorie décrite dans ce chapitre, la réponse à la première question est *la dépendance*, tandis que la réponse à la deuxième est *le pouvoir*. Nous analysons ces notions par la suite.

4.5.1 Pouvoir social

Dans [Cas90], deux notions différentes de pouvoir sont définies :

- *pouvoir de* : un agent i a le pouvoir de g s'il peut atteindre g ;

- *pouvoir sur* : un agent i a le pouvoir sur un autre agent j (en ce qui concerne g) s'il peut l'aider ou l'empêcher d'atteindre g.

Ces deux notions de pouvoir ne sont pas des concepts sociaux : elles lient un agent à un de ses buts ou bien aux buts des autres agents. Ce sont des relations *objectives*, c'est-à-dire qu'elles existent même si les agents n'en sont pas conscients.

La relation duale à celle de pouvoir est la relation de dépendance.

[6] Le formalisme que nous avons développé a l'avantage d'être implémentable, comme nous le montrons dans le chapitre 9, ce qui n'est pas le cas du modèle précédent.

4.5.2 Relations de dépendance

Un agent i est dépendant d'un agent j (en ce qui concerne le but g) s'il n'a pas le pouvoir de g et j a ce pouvoir ou bien s'il a le pouvoir de g sauf si j l'empêche d'atteindre g.

Dans [CMC92], les auteurs distinguent deux types de relations de dépendance : *dépendance de ressources* et *dépendance sociale*.

Une relation de dépendance de ressources (R-DEP) est caractérisée par les deux faits suivants :

1. un objet ou un événement dans le monde externe augmente la probabilité qu'un certain état du monde soit atteint ;
2. cet état du monde est représenté comme un but d'au moins un agent.

Dans ces conditions, nous disons que l'agent est *dépendant* de l'objet ou événement en question, ces derniers étant désignés par *ressources*.

Par ailleurs, une relation de dépendance sociale (S-DEP) est caractérisée par les deux faits suivants :

1. un agent peut entreprendre une certaine action dont le résultat augmente la probabilité qu'un certain état du monde soit atteint ;
2. cet état du monde est représenté comme un but d'un autre agent, qui pour sa part est incapable d'entreprendre l'action en question.

Les auteurs démontrent qu'une dépendance de ressource peut être réductible à une dépendance sociale si nous introduisons la notion de *contrôle* d'une ressource. De cette façon, nous pouvons imaginer qu'un agent qui contrôle une certaine ressource peut entreprendre des actions telles qu'autoriser son utilisation ou céder cette ressource à un autre agent qui en a besoin.

Les relations de dépendance présentent quelques propriétés intéressantes :

- comme les relations de pouvoir, elles sont des relations *objectives*, qui existent indépendamment du fait que les agents en soient conscients. Un agent peut ainsi dépendre d'un autre en ignorant ce fait[7] ;
- une fois que ces relations deviennent *subjectives*, plusieurs conséquences peuvent être dérivées. En particulier, elles permettent de prévoir que les agents entreprennent les actions sociales décrites dans la section 4.3.2. Nous allons mieux expliciter cette transition dans la section 4.7.

Jusqu'à présent, nous n'avons analysé que les relations de dépendance d'un agent envers un autre. Dans la section suivante, nous présentons quelques définitions supplémentaires permettant de caractériser des diverses types de relations de dépendance, concernant plusieurs agents.

[7] Nous illustrons cette situation dans le chapitre 8.

4.6 Types de relations de dépendance

Les relations de dépendance établissent un réseau social entre les agents, qui exprime d'une certaine façon le degré de dépendance et de pouvoir que les agents ont les uns par rapport aux autres. Quelques propriétés intéressantes concernant ces réseaux sont présentées dans la suite [CMC92].

4.6.1 Ou-dépendance et et-dépendance

Dans certaines situations, les relations de dépendance peuvent être composées de manière *disjonctive*. Ces situations sont modélisées par la notion de *ou-dépendance*. Nous pouvons citer au moins deux situations qui illustrent cette notion :

- *alternative de partenaires* : une action a nécessaire pour atteindre le but g peut être entreprise par un nombre d'agents différents. Il suffit que l'un d'entre eux la réalise pour que g soit atteint ;
- *alternatives d'actions* : une but g peut être atteint en entreprenant des actions différentes $a, a', \ldots$, qui peuvent être réalisées par des agents différents. Comme dans le cas précédent, il suffit qu'une de ces actions soit entreprise pour que g soit atteint.

Dans d'autres situations, les relations de dépendance peuvent être composées de façon *conjonctive*. Ces situations sont modélisées par la notion de *et-dépendance*. Nous avons aussi deux situations illustrant ce notion :

- *dépendance multi-partite* : un ensemble d'actions $a, a', \ldots$ est nécessaire pour atteindre g. Chacune d'entre elles peut être entreprise par un agent différent ;
- *dépendance multi-but* : un agent i dépend d'un autre agent j pour atteindre plusieurs buts $g, g', \ldots$, soit pour un même action soit pour des actions différentes.

Comme on le montre dans [CC95], nous pouvons affirmer qu'une ou-dépendance *diminue* le degré de dépendance/pouvoir entre deux agents, tandis qu'une et-dépendance *augmente* ce degré. Dans le cas d'une ou-dépendance, plusieurs agents lui permettant d'atteindre son but, un agent peut *choisir* le partenaire à qui s'adresser. Nous présentons un critère possible de choix dans le chapitre 7. Si, par contre, un seul agent peut l'aider à atteindre son but, intuitivement il devient plus vulnérable vis-à-vis de ce dernier. Par ailleurs, dans le cas d'une et-dépendance (plus particulièrement dans la dépendance multi-partite), un agent dépend de plusieurs autres pour atteindre son but. Il doit donc s'assurer que *tous répondront en même temps* positivement à sa requête.

4.6.2 Dépendance unilatérale et dépendance bilatérale

Nous avons décrit jusqu'à présent des situations où existe une dépendance *unilatérale*, c'est-à-dire qu'un agent i dépend d'un agent j pour atteindre son but g. Nous trouvons des situations où la dépendance est *bilatérale*, c'est-à-dire non seulement i dépend de j, mais j dépend aussi de i. Dans [Cas90] sont définis deux types de dépendance bilatérale :

- *dépendance mutuelle* : i et j dépendent l'un de l'autre pour atteindre un même but g, dont un plan pour le réaliser nécessite au moins deux actions a et a'. Chacun d'entre eux est capable d'entreprendre une de ces actions mais pas l'autre ;

- *dépendance réciproque* : i dépend de j pour atteindre son but g et j dépend de i pour atteindre son but g', les deux buts étant différents.

Selon la théorie de la dépendance, une dépendance mutuelle mène les agents à une coopération, tandis qu'une dépendance réciproque les mène à une échange social, comme nous le montrons dans la suite.

4.6.3 Dépendance externe et dépendance transitive

Dans [Con92], un modèle très intéressant de *dépendance tripartite* est présenté. Les effets non seulement des relations de dépendance mais aussi de l'adoption de buts y sont pris en compte. Prenons trois agents i, j et k. Soit g un but de i pour lequel il dépend de j. Supposons encore que i réussisse à convaincre j d'adopter son but g. De cette façon, nous pouvons définir deux types de relations de dépendance supplémentaires :

- *dépendance externe* : j dépend de k pour atteindre le but g, appartenant initialement à i, et que j a adopté ;

- *dépendance transitive* : i dépend de k pour atteindre son but g car il dépend de j et celui-ci a une dépendance externe vers k.

Normalement, la dépendance de j envers k doit être une dépendance de ressources, sinon i aurait inféré aussi une dépendance directe envers k[8].

Ce modèle de dépendance tripartite montre que si i est conscient de la dépendance externe de j envers k, il peut essayer de convaincre ce dernier de laisser j utiliser le(s) ressource(s) nécessaires pour atteindre son propre but g. Pour cela, i doit raisonner non seulement sur ses propres relations de dépendance envers j, mais aussi sur celles de j envers k.

[8] A moins que les deux agents ne représentent pas des actions primitives dans le même niveau d'abstraction.

4.7 Des relations de dépendance aux interactions sociales

Nous avons défini la notion d'autonomie au sein d'un agent, les actions sociales qu'un agent peut entreprendre et les relations de dépendance et de pouvoir. Nous sommes maintenant dans la mesure de proposer une procédure pour prédire et justifier l'interaction sociale au sein d'une société d'agents.

Les diverses étapes de cette procédure prennent en compte quelques aspects cognitifs (dans l'état mental des agents) et pré-cognitifs (observables de l'extérieur), comme le montre la figure 4.3.

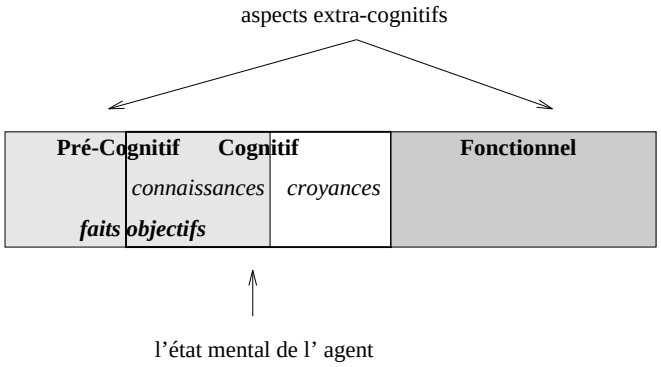


Figure 4.3 : Aspects cognitifs et pré-cognitifs de l'action sociale [CC92]

Nous pouvons prédire et justifier l'interaction sociale entre des agents par la procédure suivante :

1. les agents sont plongés dans un réseau de relations structurelles *objectives* de dépendance et de pouvoir ;

2. supposons que ces relations deviennent *subjectives*, c'est-à-dire que les agents les représentent dans leur états mentaux en tant que connaissances ;

3. une fois qu'un agent i sait qu'il dépend d'un autre agent j pour atteindre son but g, à cause d'une action a que j est capable d'entreprendre, un *but social* est créé dans son état mental : influencer j. Autrement dit, le but que j entreprenne a devient un but de i ;

4. néanmoins, comme nous l'avons montré auparavant, les relations de dépendance et de pouvoir sont en quelques sorte opposées. Normalement, si i dépend de j, c'est justement j qui a le pouvoir d'influencer i et pas le contraire. De cette façon, si i veut que j adopte son but, il doit chercher une stratégie qui puisse lui donner un certain *pouvoir d'influence* sur j, pour que l'adoption puisse être réussie. Une façon est d'attendre que l'autre agent soit bienveillant (adoption terminale) ;

5. l'adoption terminale n'étant pas le cas général, les relations de dépendance peuvent être utilisées comme une forme de pouvoir d'influence. Par exemple, si i a détecté une dépendance mutuelle envers j, il peut proposer une coopération ; s'il a inféré une dépendance réciproque, il peut essayer de lui proposer un échange social ;

6. une fois que le but social d'influencer a été généré et que les moyens pour influencer l'autre agent ont été trouvés, l'action sociale d'influence a lieu, par exemple par communication ;

7. si cette action est efficace (par exemple, j accepte de former une coalition avec i), nous pouvons justifier l'interaction sociale entre ces deux agents. Autrement dit, nous avons de nouveau un phénomène *objectif* (fonctionnel), c'est-à-dire observable à l'extérieur. Ce comportement est caractérisé comme fonctionnel dans la mesure où en nous plaçant comme observateur extérieur (et par conséquent sans accès à l'état mental des agents), nous pouvons caractériser l'interaction sociale comme une fonction des relations structurelles de dépendance et de pouvoir. Par exemple, nous pouvons dire que l'existence d'une dépendance mutuelle mène à une coopération, tandis qu'une dépendance réciproque mène à un échange social.

Nous voulons conclure cette section en ajoutant deux remarques par rapport à la procédure proposée ci-dessus. La première concerne son adéquation à la quatrième définition d'agent autonome proposée dans la section 4.2 et adoptée dans ce chapitre. Nous pouvons constater qu'aucun des principes énumérés dans notre définition n'est violé par cette procédure. Par ailleurs, l'exploitation des relations objectives dans un modèle d'interaction sociale peut paraître inhabituel. Si les agents sont conduits à agir par leurs connaissances et croyances, on peut argumenter qu'il suffit de considérer les bases cognitives pour l'interaction sociale. Dès qu'un agent a des croyances fausses, il sera de toute façon conduit à agir en conformité avec elles. La réponse à cette question est que si d'une part les croyances peuvent justifier l'action sociale, d'autre part celle-ci peut aussi justifier une révision de croyances (si elle n'a pas réussie). Nous ne voulons pas nous étendre ici sur les aspects théoriques de ce débat, une analyse plus profonde est présentée dans [CC92]. Néanmoins, dans le chapitre 8, nous montrons comment une action sociale peut être la cause de l'activation d'une révision de croyances.

4.8 Discussion

Dans ce chapitre, nous avons explicité les points principaux de la théorie de la dépendance. En particulier, nous avons montré que les dépendances bilatérales peuvent expliquer la raison pour laquelle certaines interactions sociales positives, telles que la coopération et l'échange social ont lieu.

En ce qui concerne l'objectif d'établir une théorie bien fondée sur les interactions sociales, nous avons énuméré les possibles avantages d'une approche fondée

sur la complémentarité par rapport à celle basée exclusivement sur la notion d'utilité.

Par rapport à nos objectifs, notre intention est d'utiliser les relations de dépendance bilatérales comme un critère de mesure de la possibilité qu'un agent a d'adopter un but de l'autre, comme nous le verrons plus tard dans le chapitre 7.

Finalement, nous voulons remarquer que nous ne croyons pas évidemment que l'approche utilitaire ne soit d'aucun intérêt pour modéliser les interactions sociales entre des agents. Par contre, nous contribuons à montrer que cette approche est *insuffisante* pour cette tâche. Une approche intéressante, qui mélange la représentation de buts et en quelque sens l'approche utilitaire, est présentée dans [LCD94]. Notre approche consiste aussi à essayer de fusionner ces deux perspectives, comme nous le montrons dans la suite de cette partie.

Chapitre 5

Conception du modèle

Dans ce chapitre, nous présentons le modèle formel que nous proposons pour caractériser la théorie de la dépendance. Nous avons présenté une version préliminaire de ce modèle dans [SCDC94], à laquelle nous avons ajouté quelques extensions, parmi lesquelles la notion de situation de but, présentée dans [SD95a].

Ce chapitre est organisé de la façon suivante. Tout d'abord, nous présentons les composantes que nous avons choisies pour représenter la description externe des agents dans la section 5.1. Quelques définitions préliminaires sont présentées dans la section 5.2. En utilisant ces définitions, nous décrivons en détail notre modèle dans la section 5.3, où nous présentons les idées principales de ce dernier : les *relations de dépendance*, les *situations de but* et les *situations de dépendance*. Nous terminons ce chapitre en présentant dans la section 5.4 une comparaison entre notre modèle et quelques autres aussi fondés sur la notion de dépendance.

5.1 Description externe

Comme nous l'avons exposé dans la section 3.4, un agent est caractérisé vis-à-vis des autres agents par une description externe. Par conséquent, un agent doit posséder une structure de données pour conserver ces informations sur les autres agents.

Nous pouvons trouver dans la littérature plusieurs modèles de représentation des autres [LCN90, Wer89, OCR91]. Un étude comparative de ces modèles est présenté dans [Kar95]. Dans cette étude, une comparaison détaillée entre ces modèles est présentée, en ce qui concerne les éléments qui y sont sont représentées (croyances, plans, intentions, buts, temps) ainsi qu'en relation à quelques critères comme la complexité de la représentation, sa correction et sa complétude, la faisabilité de l'implémentation etc.

Comme la théorie dont nous nous inspirons était déjà formalisée dans [CMC92], nous avons retenu ses éléments de base en ce qui concerne la représentation qu'un agent a des autres : leurs buts, leurs actions et leurs ressources. De plus, nous y avons ajouté les plans, absents de cette première formulation et qu'à notre avis

sont essentiels pour pouvoir implémenter notre mécanisme de raisonnement social.

Dans les deux sections suivantes, nous décrivons la composition de la description externe adoptée dans notre approche, ainsi que les moyens possibles qu'un agent peut utiliser pour obtenir ces informations à propos des autres.

5.1.1 Composition

Dorénavant, nous allons considérer qu'une description externe est composée de plusieurs *entrées*. Chacune de ces entrées décrit un certain agent de la société, et elles sont composées des parties suivantes :

- les *buts* qu'un agent cherche à atteindre[1] ;
- les *actions* qu'un agent est capable d'entreprendre ;
- les *ressources* sur lesquelles un agent a un certain contrôle ;
- les *plans* qu'un agent peut utiliser pour poursuivre ses buts.

Les actions et ressources utilisées dans un certain plan n'appartiennent pas nécessairement aux ensembles d'actions et de ressources propres de l'agent, il peut donc *dépendre des autres* pour exécuter un certain plan.

Par la suite, nous noterons i un agent générique dont nous voulons analyser le mécanisme de raisonnement social. Formellement, sa description externe est décrite de la façon suivante :

$$ED \triangleq \bigcup_{j=1}^{n} ED(j) \tag{5.1}$$

$$ED(j) \triangleq < id(j),\ G(j),\ A(j),\ R(j),\ P(j) > \tag{5.2}$$

$$G(j) \triangleq \bigcup_{k=1}^{n_g(j)} g_k \tag{5.3}$$

$$g_k \triangleq < id(g_k),\ w(g_k),\ s(g_k) > \tag{5.4}$$

$$A(j) \triangleq \bigcup_{k=1}^{n_a(j)} a_k \tag{5.5}$$

$$a_k \triangleq < id(a_k),\ c(a_k),\ s(a_k) > \tag{5.6}$$

$$R(j) \triangleq \bigcup_{k=1}^{n_r(j)} r_k \tag{5.7}$$

$$r_k \triangleq < id(r_k),\ c(r_k),\ s(r_k) > \tag{5.8}$$

[1] Un agent peut avoir plusieurs buts à poursuivre, mais nous ne nous intéressons pas ici à savoir si un certain but est actif ou non, cette discussion dépassant le sujet de ce chapitre.

$$P(j) \triangleq \bigcup_{k=1}^{n_p(j)} p_k \tag{5.9}$$

$$p_k \triangleq < id_g(p_k),\ I(p_k),\ s(p_k) > \tag{5.10}$$

$$I(p_k) \triangleq \bigcup_{m=1}^{n_a(p_k)} i_m \tag{5.11}$$

$$i_m \triangleq < id_a(i_m),\ R(i_m) > \tag{5.12}$$

$$R(i_m) \triangleq \bigcup_{q=1}^{n_r(i_m)} r_q^\star \tag{5.13}$$

$$r_q^\star \triangleq < id(r_q^\star) > \tag{5.14}$$

La description externe ED de l'agent i (5.1) est composée de plusieurs *entrées*. Dans chaque entrée, l'information relative à un certain agent qui appartient à la société est gardée (5.2). L'entrée $ED(j)$ relative à un agent j contient respectivement son identification $id(j)$, son ensemble de buts $G(j)$ (5.3), d'actions $A(j)$ (5.5), de ressources $R(j)$ (5.7) et de plans $P(j)$ (5.9). Ces informations correspondent à celles que l'agent i croit que l'agent j possède. Chaque plan p_k (5.10) est composé d'une identification du but $id_g(p_k)$ à atteindre, et d'une séquence d'*actions instanciées* $I(p_k)$ (5.11) utilisées dans ce plan. Une action instanciée i_m (5.12) est composée d'une identification de l'action $id_a(i_m)$ et d'une liste de ressources $R(i_m)$ (5.13) utilisées par cette action (liste pouvant être vide). Enfin, chacune de ces dernières $r_q^\star$ est composée par son identification $id(r_q^\star)$ (5.14).

Pour chaque but g_k (5.4), action a_k (5.6), ressource r_k (5.8) et plan p_k (5.10), nous définissons aussi une fonction qui permet d'obtenir la *source* de cette information, dénotée respectivement par $s(g_k)$, $s(a_k)$, $s(r_k)$ et $s(p_k)$. Cet aspect est discuté en plus de détail dans la section suivante.

Finalement, pour les buts, actions et ressources, nous définissons aussi respectivement un ensemble de fonctions $w(g_k)$, $c(a_k)$ et $c(r_k)$ pour mesurer l'importance d'un but et les coûts d'une action ou ressource. Ces informations peuvent être utilisés par le mécanisme de décision d'un agent quand celui-ci doit choisir un but à poursuivre ou un plan à exécuter, comme nous le montrons dans le chapitre 7. Dans une première approche, nous allons supposer que *tous les agents évaluent de la même façon les coûts des actions et des ressources*. Nous sommes conscients qu'une telle hypothèse est d'une part assez restrictive, mais d'autre part elle nous permet de séparer clairement les choix de plans du choix de partenaires. Cette séparation nous est utile, puisque nous intéressons plutôt par la problématique du choix de partenaires dans le cadre de nos recherches.

Pour illustrer le concept de description externe, prenons comme exemple une société composée de quatre agents : $ag1$, $ag2$, $ag3$ et $ag4$. Supposons que l'agent $ag1$ croit qu'un but de l'agent $ag3$ soit de rentrer chez lui, ce qui peut être fait en utilisant sa voiture. Nous présentons dans la figure 5.1 l'entrée de la description

externe relative à l'agent $ag3$, appartenant à l'agent $ag1$.

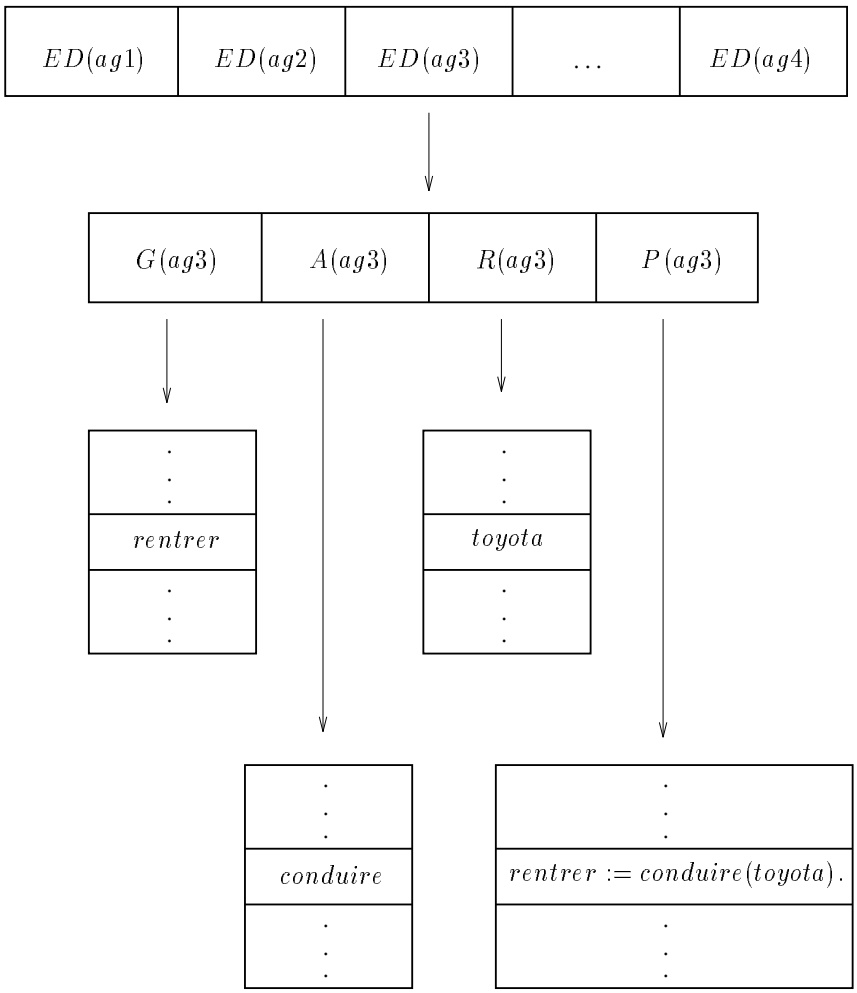


Figure 5.1 : Description externe

5.1.2 Sources d'information

Selon le modèle présenté dans la section 3.4, une information peut être acquise par un agent à partir de trois sources différentes : réception d'un message[2], perception et inférence, cette dernière correspondant à son mécanisme de raisonnement. Par exemple, si un agent reçoit un message d'un autre agent où ce dernier affirme être né au Brésil et être capable de lire le français, il peut déduire que ce dernier est capable aussi d'entreprendre l'action de traduire du français en portugais.

[2]Il s'agit plus précisément de l'identification de l'agent qui a envoyé le message.

Nous considérons que ces informations peuvent être initialisées et mises à jour dynamiquement. Une proposition présentée dans [BDB92] est d'offrir aux agents la possibilité d'utiliser un *protocole d'introduction* dans la société, et donc ce processus d'acquisition est réalisé par communication explicite. Nous utilisons cette approche dans la conception du système DEPINT, présenté dans le chapitre 11.

Nous considérons que de façon générale *les informations que les agents ont les uns sur les autres ne sont pas nécessairement ni complètes ni correctes.* Si nous analysons les sources d'acquisition de ces informations, nous pouvons remarquer les faits suivants :

- les mécanismes de perception sont souvent sujets à des erreurs ;
- les agents peuvent, dans certaines conditions, ne pas communiquer toutes ses capacités aux autres[3] ;
- les mécanismes de raisonnement d'un agent peuvent avoir des règles incorrectes. Dans l'exemple présenté au début de cette section, l'agent a utilisé implicitement une règle permettant de déduire que celui qui est né au Brésil parle le portugais. Cette règle n'est pas tout à fait correcte : une personne peut être née au Brésil et avoir quittée ce pays depuis l'âge d'un an, et dans ce cas-là elle probablement ne parle pas le portugais.

Si nous considérons que ces mécanismes d'acquisition sont imparfaits, la source d'une information peut devenir important quand un agent doit deviser ses croyances, par exemple quand deux agents détectent qu'ils ont des croyances incomplètes ou incorrectes l'un sur l'autre, comme nous le montrons dans le chapitre 8.

5.2 Définitions préliminaires

Dans la section 5.2.1, nous introduisons les définitions d'agent *sujet*, agent *objet*, agent *tiers* et agent *source* pour désigner de façon précise les différents rôles qu'un agent peut jouer dans le mécanisme de raisonnement social. Ensuite, nous expliquons dans la section 5.2.2 la raison pour laquelle nous avons explicitement choisi de faire une différenciation entre les notions d'autonomie/de dépendance d'actions et de ressources.

5.2.1 Agents sujet, objet, tiers et source

Comme nous allons montrer par la suite, les différentes notions de dépendance et d'autonomie sont strictement liées à *l'ensemble des plans utilisés dans le raisonnement.* Comme nous l'avons montré dans la section 5.1, un agent représente les

[3] Nous verrons plus tard dans le chapitre 6 que la génération endogène de buts et la planification justifient cette affirmation.

buts, actions, ressources et plans des autres agents, et par conséquent, il peut se servir des plans d'autrui pour inférer diverses propriétés, même si cela n'est pas le cas général. Néanmoins, dans un cadre strictement formel, il est intéressant de pouvoir représenter la situation la plus générale possible.

Ayant cette démarche en tête, nous allons donc utiliser les termes suivants pour désigner de façon précise des différents rôles qu'un agent peut jouer dans notre mécanisme de raisonnement social :

agent sujet : c'est l'agent qui est en train de raisonner ;

agent objet : c'est l'agent sur lequel une inférence est faite ;

agent tiers : c'est l'agent sur lequel l'agent objet a une relation de dépendance ;

agent source : c'est l'agent dont les plans sont utilisés par l'agent sujet pour inférer des différentes propriétés relatives à l'agent objet.

Pour illustrer ces notions, reprenons la société présentée dans la section 5.1. Supposons encore que l'agent $ag1$ croit que selon les plans de l'agent $ag4$, celui qui veut écrire une lettre adressée à la Belgique flamande doit nécessairement mettre l'adresse sous l'enveloppe en flamand. L'agent $ag1$ sait que : (i) l'agent $ag3$ doit envoyer une lettre importante à un partenaire qui habite Ostende, (ii) ce dernier ne parle pas flamand et (iii) l'agent $ag2$, étant originaire de la Belgique flamande, parle flamand. Supposons que l'agent $ag1$ dit à l'agent $ag3$: "Il faut que tu demandes à ag2 comment s'écrit ton adresse en flamand, car autrement ta lettre n'arriverait jamais". Dans cet exemple, nous avons que l'agent $ag1$ est l'agent *sujet* (celui qui raisonne), l'agent *objet* est l'agent $ag3$ (l'agent sur lequel l'agent $ag1$ raisonne), l'agent *tiers* est l'agent $ag2$ (l'agent de qui l'agent $ag3$ dépend) et l'agent *source* est l'agent $ag4$ (l'agent dont les plans sont ceux que l'agent $ag1$ a utilisé pour raisonner).

Dans le cadre de la section 5.3, où nous décrivons le langage interne des agents, *l'agent sujet n'est pas explicitement représenté*, car nous ne supposons pas dans ce contexte que l'agent qui raisonne a la propriété d'introspection. Nous allons néanmoins le représenter explicitement dans le chapitre 8, où nous nous intéressons à l'analyse des résultats obtenus par les mécanismes de raisonnement social de deux agents différents. Par conséquent, tout au long de ce chapitre, les phrases telles que "les plans qui appartiennent à l'agent source k" doivent être interprétés comme *les plans que l'agent sujet croit appartenir à l'agent source* k, c'est-à-dire, les plans qui appartiennent à l'entrée de la description externe relative à l'agent k, cette description appartenant à l'agent sujet.

5.2.2 Types d'autonomie et types de dépendance

Pour une question de uniformisation du langage avec les travaux sur lesquels notre modèle est fondé [Cas90, CMC92], nous allons utiliser tout au long de ce

chapitre le terme *autonomie* comme un synonyme d'*auto-suffisance*.

En considérant notre modèle de description externe présenté dans la section 5.1, nous avons explicitement différencié les actions des ressources utilisées dans un plan. Ce choix permet de représenter de façon distincte le fait qu'un agent puisse entreprendre toutes les actions d'un certain plan du fait qu'il a le contrôle sur toutes les ressources utilisées par ce même plan. Du point de vue strictement cognitif, nous croyons que cette distinction est utile. Supposons, par exemple, qu'un agent a besoin de traduire un mot du français à l'anglais, et qu'il connaît un autre agent qui parle couramment ces deux langues et qui a un dictionnaire français-anglais. Le premier agent peut atteindre son but de deux façons différentes : (i) il peut demander au deuxième de traduire le mot ou (ii) il peut demander au deuxième de lui prêter le dictionnaire, pour qu'il puisse lui-même chercher la traduction du mot. En ce qui concerne le deuxième agent, dans la première option il doit arrêter son travail pour faire la traduction, tandis que simplement emprunter au premier le dictionnaire ne lui empêche pas de continuer ses activités courantes. De cette façon, nous croyons qu'*entreprendre une action* engage plus un agent que simplement *laisser un autre utiliser ses ressources*. Pour cette raison nous avons décidé de traiter différemment les notions de autonomie/dépendance d'actions/de ressources, qui seront détaillées respectivement dans les sections 5.3.5 et 5.3.6.

5.3 Langage interne

Nous supposons que le langage interne des agents utilisé par le mécanisme de raisonnement social soit un langage de premier ordre.

Comme nous voulons déduire certaines propriétés concernant les buts, plans, actions et ressources, toutes les variables utilisées sont *typées*. Nous allons utiliser la convention suivante :

- $\{i, j, k\}$: variables qui désignent des agents ;
- $\{g, g'\}$: variables qui désignent des buts ;
- p: variable qui désigne des plans ;
- a: variable qui désigne des actions ;
- r: variable qui désigne des ressources.

5.3.1 Notions de base

Nous proposons les prédicats suivants, qui expriment des notions élémentaires concernant des buts, des plans, des actions et des ressources :

- $achieves(p, g)$ exprime le fait que le plan p atteint le but g ;

- $uses_a(p, a)$ exprime le fait que l'action a est nécessaire pour exécuter le plan p ;
- $uses_r(p, r)$ exprime le fait que la ressource r est nécessaire pour exécuter le plan p ;
- $diff(g, g')$ exprime le fait que les buts g et g' sont différents.

5.3.2 Interfaçage avec la description externe

Nous avons besoin de définir quelques prédicats qui expriment le fait que l'agent sujet *croît* que l'agent i a un certain but g, qu'il peut entreprendre une certaine action a, qu'il a le contrôle sur une certaine ressource r ou qu'il a un certain plan p. Pour cela, nous définissons les prédicats suivants :

$$is_g(i, g) \triangleq \{\exists ED(t) \in ED \mid (id(t) = i \wedge \exists g_k \in G(t) \mid id(g_k) = g)\} \quad (5.15)$$

$$is_a(i, a) \triangleq \{\exists ED(t) \in ED \mid (id(t) = i \wedge \exists a_k \in A(t) \mid id(a_k) = a)\} \quad (5.16)$$

$$is_r(i, r) \triangleq \{\exists ED(t) \in ED \mid (id(t) = i \wedge \exists r_k \in R(t) \mid id(r_k) = r)\} \quad (5.17)$$

En ce qui concerne les plans, la définition est un peu plus compliquée, car l'identité d'un plan est définie de façon récursive à partir du but qu'il est censé atteindre, ses actions instanciées et les ressources utilisées par ces dernières :

$$\begin{aligned} is_p(i, p) \triangleq \{&\exists ED(t) \in ED \mid (id(t) = i \wedge \exists p_k \in P(t) \mid \\ &(\exists g\ achieves(p, g) \mid id_g(p_k) = g \wedge \forall i_m \in I(p_k) \\ &(\exists a\ uses_a(p, a) \mid id_a(i_m) = a \wedge \forall r_q^\star \in R(i_m) \\ &(\exists r\ uses_r(p, r) \mid id(r_q^\star) = r))))\} \end{aligned} \quad (5.18)$$

La partie droite des équations ci-dessus décrit la liaison entre ces prédicats et la structure correspondant à la description externe définie dans la section 5.1.

5.3.3 Notions auxiliaires

Nous allons définir aussi quelques prédicats pour exprimer certaines notions auxiliaires dont nous aurons besoin pour formuler notre théorie :

- $is\_plan(i, g, p)$ exprime le fait que p est un plan qui appartient à l'agent i et qui atteint le but g ;
- $has\_plans(i, g)$ exprime le fait que l'agent l'agent i a au moins un plan pour atteindre le but g ;
- $needs_a(i, p, a)$ exprime le fait que l'agent i a besoin de l'aide d'un autre agent pour entreprendre l'action a qui est utilisée dans le plan p ;

- $needs_r(i,p,r)$ exprime le fait que l'agent i a besoin de de la ressource r sur laquelle il n'a pas de contrôle pour exécuter le plan p ;
- $has\_all_a(i,p)$ exprime le fait que l'agent i peut entreprendre toutes les actions nécessaires pour exécuter le plan p ;
- $has\_all_r(i,p)$ exprime le fait que l'agent i a le contrôle sur toutes les ressources nécessaires pour exécuter le plan p.

Formellement, nous avons :

$$is\_plan(i,g,p) \Leftrightarrow is_p(i,p) \wedge achieves(p,g) \quad (5.19)$$
$$has\_plans(i,g) \Leftrightarrow \exists p\ is\_plan(i,g,p) \quad (5.20)$$
$$needs_a(i,p,a) \Leftrightarrow uses_a(p,a) \wedge \neg is_a(i,a) \quad (5.21)$$
$$has\_all_a(i,p) \Leftrightarrow \forall a\ (uses_a(p,a) \Rightarrow is_a(i,a)) \quad (5.22)$$
$$needs_r(i,p,r) \Leftrightarrow uses_r(p,r) \wedge \neg is_r(i,r) \quad (5.23)$$
$$has\_all_r(i,p) \Leftrightarrow \forall r\ (uses_r(p,r) \Rightarrow is_r(i,r)) \quad (5.24)$$

5.3.4 Notions sur les plans

Après avoir défini les notions présentées ci-dessus, nous pouvons maintenant formuler les notions de *plan autonome* et *plan dépendant*.

Un plan p rend l'agent objet i *a-autonome* pour le but g si et seulement si cet agent peut entreprendre toutes les actions nécessaires pour son exécution (prédicat aut_plan_a). De façon analogue, nous pouvons définir par rapport aux ressources la notion de plan *r-autonome* nécessaires pour son exécution (prédicat aut_plan_r). De façon. Dans la représentation formelle présentée ci-dessous, nous indiquons aussi l'agent source k auquel ce plan appartient :

$$aut\_plan_a(i,g,p,k) \Leftrightarrow is\_plan(k,g,p) \wedge has\_all_a(i,p) \quad (5.25)$$
$$aut\_plan_r(i,g,p,k) \Leftrightarrow is\_plan(k,g,p) \wedge has\_all_r(i,p) \quad (5.26)$$

Par ailleurs, un plan p rend l'agent objet i *a-dépendant* de l'agent tiers j pour le but g si et seulement si dans ce plan il existe au moins une action que l'agent i ne peut pas entreprendre et que l'agent j peut l'entreprendre (prédicat dep_plan_a). De façon analogue, nous définissons par rapport aux ressources la notion de plan *r-dépendant* (prédicat dep_plan_r). Une fois de plus, nous indiquons aussi l'agent source k auquel ce plan appartient dans la représentation formelle :

$$dep\_plan_a(i,j,g,p,k) \Leftrightarrow is_p(k,p) \wedge \exists a\ basic\_dep_a(i,j,g,p,a) \quad (5.27)$$
$$dep\_plan_r(i,j,g,p,k) \Leftrightarrow is_p(k,p) \wedge \exists r\ basic\_dep_r(i,j,g,p,r) \quad (5.28)$$

où :

$$basic\_dep_a(i,j,g,p,a) \Leftrightarrow achieves(p,g) \wedge needs_a(i,p,a) \wedge is_a(j,a) \quad (5.29)$$
$$basic\_dep_r(i,j,g,p,r) \Leftrightarrow achieves(p,g) \wedge needs_r(i,p,r) \wedge is_r(j,r) \quad (5.30)$$

Les prédicats $basic\_dep_a$ et $basic\_dep_r$ représentent les *dépendances de base* concernant respectivement les actions et les ressources. Par exemple, le prédicat $basic\_dep_a(i, j, g, p, a)$ exprime que (i) p est un plan qui atteint g, (ii) selon le plan p, il faut entreprendre l'action a pour atteindre ce but, (iii) i est incapable d'entreprendre l'action a et (iv) j est capable de entreprendre cette action. Ces prédicats correspondent d'une certain façon à une extension des définitions originales de S-DEP et R-DEP introduites dans la section 4.5.2 et formellement présentées dans [CMC92]. Nous avons modifiés ces derniers selon deux aspects : (i) nous avons représenté la notion de plan qui était absente de la formulation originale et (ii) nous avons représenté une dépendance d'action par l'indice a (à la place de S-DEP). Nous allons utiliser dans la suite l'indice s pour désigner une dépendance soit d'action soit de ressources, que nous appelons une dépendance sociale.

5.3.5 Notions d'autonomie

Maintenant, nous pouvons définir les différentes notions d'autonomie qui nous intéressent. Un agent objet i est *a-autonome* pour le but g *en considérant les plans appartenant à l'agent source* k (prédicat aut_a) si et seulement si : (i) i a le but g ; (ii) k a un plan qui rend i a-autonome pour g. De façon analogue, nous définissons par rapport aux ressources la notion de *r-autonomie* (prédicat aut_r). Si un agent objet i est à la fois a-autonome et r-autonome pour g, alors i est dit *s-autonome* (prédicat aut_s) pour le but g :

$$aut_a(i,g,k) \Leftrightarrow is_g(i,g) \wedge has\_plans(k,g) \wedge \exists p\ aut\_plan_a(i,g,p,k) \quad (5.31)$$

$$aut_r(i,g,k) \Leftrightarrow is_g(i,g) \wedge has\_plans(k,g) \wedge \exists p\ aut\_plan_r(i,g,p,k) \quad (5.32)$$

$$aut_s(i,g,k) \Leftrightarrow aut_a(i,g,k) \wedge aut_r(i,g,k) \quad (5.33)$$

5.3.6 Relations de dépendance

Si un agent objet i n'est pas autonome pour un but g, il est *dépendant des autres* pour ce but. Nous définissons les notions de *a-dépendance* (prédicat dep_a), *r-dépendance* (prédicat dep_r) et *s-dépendance* (prédicat dep_s) de la façon suivante :

$$dep_a(i,g,k) \Leftrightarrow is_g(i,g) \wedge has\_plans(k,g) \wedge \neg\exists p\ aut\_plan_a(i,g,p,k) \quad (5.34)$$

$$dep_r(i,g,k) \Leftrightarrow is_g(i,g) \wedge has\_plans(k,g) \wedge \neg\exists p\ aut\_plan_r(i,g,p,k) \quad (5.35)$$

$$dep_s(i,g,k) \Leftrightarrow dep_a(i,g,k) \vee dep_r(i,g,k) \quad (5.36)$$

Le fait qu'un agent soit dépendant des autres pour un certain but ne signifie pas a priori qu'il existe un agent dans la société qui puisse entreprendre l'action ou lui donner le contrôle sur la ressource dont il a besoin. Nous devons introduire une deuxième notion de dépendance, que nous appelons *relation de dépendance*, cette fois-ci en introduisant explicitement la notion d'agent tiers défini dans la section 5.2.1.

Un agent objet i est dit *a-dépendant* d'un agent tiers j pour le but g *en considérant les plans appartenant à l'agent source* k (prédicat dep_on_a) si et seulement si : (i) i a le but g ; (ii) i est a-dépendant pour g en considérant les plans de k ; (iii) k a un plan qui rend i a-dépendant de j. Les autres définitions sont construites de façon analogue, comme dans les sections précédentes (prédicats dep_on_r et dep_on_s) :

$$dep\_on_a(i,j,g,k) \Leftrightarrow dep_a(i,g,k) \wedge \exists p\ dep\_plan_a(i,j,g,p,k) \quad (5.37)$$

$$dep\_on_r(i,j,g,k) \Leftrightarrow dep_r(i,g,k) \wedge \exists p\ dep\_plan_r(i,j,g,p,k) \quad (5.38)$$

$$dep\_on_s(i,j,g,k) \Leftrightarrow dep\_on_a(i,j,g,k) \vee dep\_on_r(i,j,g,k) \quad (5.39)$$

5.3.7 Dépendance mutuelle et dépendance réciproque

Lorsqu'un agent sujet i déduit qu'il dépend[4] de l'agent tiers j, il est plutôt intéressant à lui de savoir si cette dépendance est unilatérale ou bilatérale.

D'après les définitions introduites dans la section 4.6.2, nous appelons *dépendance mutuelle* (MD) la situation où l'agent objet i et l'agent tiers j sont a-dépendants l'un de l'autre *pour un même but* g. Par ailleurs, nous appelons *dépendance réciproque* (RD) la situation où ces deux agents a-dépendent l'un de l'autre, mais *pour des buts différents* g et g'. Formellement, toujours *en considérant l'agent* k *comme la source des plans*, nous avons :

$$MD(i,j,g,k) \Leftrightarrow dep\_on_a(i,j,g,k) \wedge dep\_on_a(j,i,g,k) \quad (5.40)$$

$$RD(i,j,g,g',k) \Leftrightarrow dep\_on_a(i,j,g,k) \wedge dep\_on_a(j,i,g',k) \wedge diff(g,g') \quad (5.41)$$

Rappelons que ces deux notions sont strictement liées aux interactions sociales de *coopération* et d'*échange social*, comme nous l'avons montré dans la section 4.7. Nous voulons aussi remarquer que nous n'avons utilisé que les dépendances d'actions pour définir ces prédicats. Nous justifions ce fait dans la suite.

5.3.8 Quelques simplifications

A fin de continuer la construction de notre modèle, nous avons adopté deux hypothèses simplificatrices, décrites dans la suite.

Contraintes sur les agents sujet, objet, tiers et source

L'idée principale de ce travail réside dans le fait que les diverses notions d'autonomie et de dépendance sont *fortement liées à l'ensemble des plans utilisés dans le mécanisme de raisonnement.* Autrement dit, un agent sujet peut utiliser soit *ses propres* plans (dans ce cas, l'agent sujet et l'agent source désignent le même agent) soit *ceux des autres* pour déduire sa possible autonomie ou dépendance pour un certain but. Dans ce dernier cas, nous pouvons considérer

[4]Dans cette situation, l'agent objet et l'agent sujet désignent le même agent.

que l'agent sujet fait un espèce de simulation du mécanisme de raisonnement social de l'agent source, cette même approche étant exploitée dans un cadre logique dans [Wai94a].

Par ailleurs, en ce qui concerne un agent sujet, le fait de ne pas contraindre ses possibles agents sources pour calculer ses dépendances a un effet de bord indésirable : la complexité combinatoire du calcul. Ayant en tête que nos objectifs ne se résument pas seulement à proposer un cadre théorique pour le mécanisme de raisonnement social, mais de l'implémenter de la façon la plus efficace possible, nous avons fixé les contraintes suivantes sur ce calcul, en considérant toujours un même agent sujet qui est en train de raisonner :

1. l'agent initialement ne calcule que ses propres relations de dépendance envers des autres, en utilisant ses propres plans. Autrement dit, nos notions d'agents sujet, objet et source désignent le même agent ;

2. après avoir détecté une relation de dépendance envers un agent tiers, l'agent calcule les possibles relations de dépendance de ce dernier envers lui. Autrement dit, les agents qui tiennent lieu d'agent objet et d'agent tiers sont inter-changés. Par contre, l'agent utilise toujours ses propres plans, c'est-à-dire, l'agent source désigne toujours l'agent sujet lui-même ;

3. dans le cas où une dépendance mutuelle ou réciproque est détectée à l'étape précédente, l'agent essaie de vérifier si cette même conclusion pourrait être inférée en utilisant les plans de l'agent tiers. Autrement dit, l'agent qui tenait lieu d'agent tiers dans la première étape devient celui qui tient lieu d'agent source, et on répète les deux étapes précédentes.

Nous croyons qu'en fixant ces contraintes, nous pouvons diminuer la complexité de calcul des diverses propriétés, en maintenant en même temps les aspects essentiels de notre modèle, c'est-à-dire la possibilité pour un agent sujet d'inférer quelques propriétés qui selon lui puissent être aussi déduites par des autres agents. Nous allons mieux exploiter cet aspect en introduisant la notion de situation de dépendance dans la section 5.3.10.

Contraintes sur les types de dépendance

Comme nous l'avons expliqué dans la section 5.2.2, nous croyons qu'une relation de dépendance d'actions n'a pas la même signification qu'une dépendance de ressources. C'est pour cette raison, d'ailleurs, que dans notre formulation de dépendance mutuelle et réciproque, présentée dans les équations (5.40) et (5.41) nous n'avons utilisé que les dépendances d'actions. Par exemple, nous aurions pu utiliser la notion de s-dépendance, présentée dans l'équation (5.36) pour caractériser ces premières, mais à notre avis il n'y a pas de symétrie entre ces deux notions : nous voulons distinguer la situation où $dep\_on_a(i,j,g,k) \wedge dep\_on_r(j,i,g,k)$ de la

situation où $dep\_on_r(i,j,g,k) \wedge dep\_on_a(j,i,g,k)$. Nous avons discuté une exploitation possible de cette différenciation lorsque nous avons introduit les notions de dépendance externe et dépendance transitive dans la section 4.6.3.

Nous croyons que cette distinction est importante et doit être analysée plus profondément dans nos travaux futurs. Néanmoins, le fait de la prendre en compte dans ce travail aurait généré deux effets indésirables : (i) une complication inutile du notre modèle et (ii) la déviation de son idée centrale, c'est-à-dire, montrer comment un mécanisme de raisonnement social peut être exploité par un agent. De cette façon, dans nos formulations suivantes, nous allons nous limiter à l'analyse de dépendances d'actions, et dorénavant nous n'allons plus utiliser le préfixe pour désigner le type de dépendance, c'est-à-dire, nous allons utiliser les termes *autonome* et *dépendant* comme des synonymes de *a-autonome* et *a-dépendant*.

5.3.9 Situations de but

Nous appelons *situation de but* une notion liant un agent à un but quelconque, qui a quatre possibilités :

- **but inexistant (NG)** : l'agent n'a pas ce but dans son ensemble de buts à atteindre ;
- **plan inexistant (NP)** : l'agent veut atteindre ce but, mais il n'y a aucun plan permettant de l'atteindre ;
- **autonome (AUT)** : l'agent veut atteindre ce but, il a des plans qui lui permettent de l'atteindre et un de ces plans le rend autonome pour ce but ;
- **dépendant (DEP)** : l'agent veut atteindre ce but, il a des plans qui lui permettent de l'atteindre mais tous ces plans le rendent dépendant pour ce but.

Formellement, nous avons :

$$NG(i,g) \quad \Leftrightarrow \quad \neg is_g(i,g) \tag{5.42}$$
$$NP(i,g) \quad \Leftrightarrow \quad is_g(i,g) \wedge \neg has\_plans(i,g) \tag{5.43}$$
$$AUT(i,g) \quad \Leftrightarrow \quad aut_a(i,g,i) \tag{5.44}$$
$$DEP(i,g) \quad \Leftrightarrow \quad dep_a(i,g,i) \tag{5.45}$$

Normalement, étant donné un certain but à accomplir, un agent calcule d'abord sa situation de but. S'il est dépendant pour ce but ($DEP(i,g)$), il va calculer sa situation de dépendance envers des autres agents pour ce but, comme nous montrons par la suite.

5.3.10 Situations de dépendance

En étant dépendant pour un certain but, et après avoir calculé ses relations de dépendances, un agent peut utiliser ces dernières pour raisonner sur les autres. En particulier, pour un certain but g, un agent i peut calculer sa *situation de dépendance* envers chaque agent j appartenant à la société.

Un agent déduit une *dépendance localement reconnue* ("locally believed dependence"), soit mutuelle soit réciproque, s'il utilise exclusivement *ses propres plans* en raisonnant sur les autres. S'il utilise à la fois *ses propres plans* et *ceux de l'agent tiers* duquel il dépend pour arriver à cette conclusion, on parlera plutôt de *dépendance mutuellement reconnue* ("mutually believed dependence") entre eux[5].

Considérons deux agents i et j, dont l'agent sujet est i. Si i infère $DEP(i, g)$ comme situation de but pour g, il existe six situations de dépendance différentes envers l'agent tiers j qui peuvent survenir, en considérant son mécanisme de raisonnement social :

1. **Indépendance (IND)** : utilisant ses propres plans, i déduit qu'il ne dépend pas de j pour g ;
2. **Dépendance Mutuelle Localement Reconnue (LBMD)** : utilisant ses propres plans, i déduit une dépendance mutuelle envers j, mais il ne peut pas déduire la même conclusion en utilisant les plans de j ;
3. **Dépendance Mutuelle Mutuellement Reconnue (MBMD)** : utilisant ses propres plans, i déduit une dépendance mutuelle envers j. En outre, en utilisant les plans de j, il peut déduire la même conclusion ;
4. **Dépendance Réciproque Localement Reconnue (LBRD)** : utilisant ses propres plans, i déduit une dépendance réciproque envers j, mais il ne peut pas déduire la même conclusion en utilisant les plans de j ;
5. **Dépendance Réciproque Mutuellement Reconnue (MBRD)** : utilisant ses propres plans, i déduit une dépendance réciproque envers j. En outre, en utilisant les plans de j, il peut déduire la même conclusion ;
6. **Dépendance Unilatérale (UD)** : utilisant ses propres plans, i déduit qu'il dépend de j pour g, mais ce dernier ne dépend pas de lui pour aucun de ses buts.

Formellement, nous avons :

$$IND(i, j, g) \quad \Leftrightarrow \quad DEP(i, g) \wedge \neg dep\_on_a(i, j, g, i) \tag{5.46}$$

[5]Nous sommes conscients que dans notre contexte, le terme "mutually believed" ne désigne pas la notion de "mutual belief" souvent trouvée dans la littérature de SMA, comme par exemple dans [LCN90]. Pour nous, ce terme désigne le fait que l'agent sujet croit que l'agent tiers duquel il dépend est aussi au courant de leurs relations de dépendance l'un envers l'autre.

$$LBMD(i,j,g) \Leftrightarrow MD(i,j,g,i) \wedge \neg MD(i,j,g,j) \quad (5.47)$$

$$MBMD(i,j,g) \Leftrightarrow MD(i,j,g,i) \wedge MD(i,j,g,j) \quad (5.48)$$

$$LBRD(i,j,g,g') \Leftrightarrow RD(i,j,g,g',i) \wedge \neg RD(i,j,g,g',j) \quad (5.49)$$

$$MBRD(i,j,g,g') \Leftrightarrow RD(i,j,g,g',i) \wedge RD(i,j,g,g',j) \quad (5.50)$$

$$UD(i,j,g) \Leftrightarrow dep\_on_a(i,j,g,i) \wedge \neg\exists g' \, (is_g(j,g') \wedge dep\_on_a(j,i,g',i)) \quad (5.51)$$

Il faut considérer que dans les situations de dépendance mutuellement reconnues, à aucun moment nous n'avons supposé que *les plans des agents devaient être identiques*. Ce point a l'avantage de ne pas restreindre inutilement le modèle proposé.

Nous allons montrer dans le chapitre 7 que les situations de dépendance peuvent être exploitées par un agent quand ce dernier doit choisir les partenaires à qui envoyer une proposition de formation de coalition, lorsqu'il est incapable d'atteindre un de ses buts tout seul.

5.4 Discussion

Pendant ces dernières années, les méthodes d'analyse fondées sur les réseaux structurels sont utilisées de plus en plus souvent dans les domaines de sciences sociales et politiques [Kno90]. Néanmoins, ces méthodes supposent toujours l'existence d'un observateur extérieur qui modélise ces relations objectives entre des agents. A notre connaissance, aucune méthode n'a encore proposé d'étudier l'influence de l'aspect subjectif de ces relations, par exemple comment le fait qu'un agent connaissant plus ou moins ces relations modifie son propre comportement et plus généralement celui de la société. Nous croyons que selon cet aspect, nos travaux sont assez originaux, car nous en prenons compte de deux façons : (i) chaque agent a sa propre représentation de ces relations, inférées à partir de sa propre description externe et (ii) chaque agent prend en compte le fait qu'un autre agent puisse être conscient ou non de ces mêmes relations, en utilisant les notions de dépendance localement ou mutuellement reconnue.

Dans le domaine de l'informatique, les relations de dépendance commencent aussi à connaître une utilisation de plus en plus répandue. Néanmoins, la plupart des travaux ne concernent que la *phase de conception du système*. Dans ce contexte, il s'agit d'utiliser ces relations pour concevoir les liens structurels/organisationnels entre un certain nombre d'agents conçus pour résoudre un problème particulier. Comme exemple de ces approches, nous pouvons citer la méthode CASSIOPÉE, inspirée dans nos travaux [CCZ94, page 4], et les travaux présentés dans [YM93] et [LCD95]. Aucun de ces travaux n'aborde directement l'aspect subjectif de la représentation des relations de dépendance.

Bien que nos intérêts scientifiques présentés dans la section 1.2 ne concernent cet aspect, nous montrons dans nos perspectives présentées dans le chapitre 12 une exploitation possible de notre modèle dans ce sens-là. Nous avons même réalisé

quelques études préliminaires concernant cet aspect, qui sont présentées dans les annexes B et C, où nous illustrons l'utilisation de notre modèle comme un outil formel pour décrire les organisations d'une société d'agents.

Finalement, dans [Par90] nous trouvons un modèle formel de contrôle entre agents fondé sur les notions de dépendance et de pouvoir. Ce modèle est assez proche de celui proposé dans [CMC92]. Dans ce travail, l'auteur reconnaît qu'une notion de plan est nécessaire pour mieux développer sa théorie. C'est justement cette notion que nous avons ajoutée au modèle initialement proposé dans [CMC92].

Chapitre 6

Effet sur l'adaptation

Dans ce chapitre, nous analysons l'effet du mécanisme de raisonnement social sur l'*adaptation* d'un agent dans un contexte d'un SMA ouvert. En particulier, nous montrons qu'un agent peut utiliser certains résultats inférés par ce mécanisme pour choisir un but à poursuivre ou un plan à exécuter, de façon à s'assurer que toutes les fonctionnalités nécessaires pour l'exécution du plan sont disponibles dans la société.

Ce chapitre est organisé de la façon suivante. Tout d'abord, nous introduisons dans la section 6.1 certaines définitions supplémentaires à notre modèle pour exprimer qu'un plan est *exécutable* ou non et qu'un but est *réalisable* ou non. En utilisant ces définitions, nous montrons respectivement dans les sections 6.2 et 6.3 comment le mécanisme de raisonnement social peut aider un agent dans ses choix de but et de plan. Pour chacun de ces choix, nous décrivons la procédure de choix, un critère formel qui l'exprime et la gestion du choix par le modèle d'agent. Enfin, nous terminons ce chapitre en présentant dans la section 6.4 une discussion sur les résultats obtenus.

6.1 Plans exécutables et buts réalisables

Nous plaçant dans le contexte d'une société ouverte, nous ne pouvons pas garantir que les plans qu'un agent a pour atteindre un certain but soient toujours *exécutables*. L'exécutabilité d'un plan implique que toutes les actions/ressources nécessaires à son exécution sont couramment disponibles dans la société d'agents, c'est-à-dire qu'il existe au moins un agent qui peut entreprendre chacune de ces actions ou qui contrôle chacune de ces ressources.

Nous introduisons ainsi deux prédicats qui expriment respectivement qu'une action ou une ressource est *disponible* (prédicat *available*) dans la société à un instant donné :

$$available_a(a) \Leftrightarrow \exists i \; is_a(i, a) \tag{6.1}$$

$$available_r(r) \Leftrightarrow \exists i \; is_r(i, r) \tag{6.2}$$

En utilisant ces définitions, nous pouvons représenter trois différentes notions de l'exécutabilité d'un plan (prédicat $feasible$), comme nous avons fait pour les notions d'autonomie et de dépendance dans les sections 5.3.5 et 5.3.6. Intuitivement, un plan est *a-exécutable* si toutes les actions nécessaires à son exécution peuvent être entreprises par au moins un agent appartenant à la société. De façon analogue, nous définissons par rapport aux ressources la notion de plan *r-exécutable*. Un plan qui est à la fois a-exécutable et r-exécutable est dit *s-exécutable* :

$$feasible_a(p) \Leftrightarrow \forall a\ (uses_a(p,a) \Rightarrow available_a(a)) \quad (6.3)$$
$$feasible_r(p) \Leftrightarrow \forall r\ (uses_r(p,r) \Rightarrow available_r(r)) \quad (6.4)$$
$$feasible_s(p) \Leftrightarrow feasible_a(p) \wedge feasible_r(p) \quad (6.5)$$

De la même manière, nous ne pouvons pas garantir non plus que les buts d'un agent soient toujours *réalisables*. Un but sera *réalisable* (prédicat *achievable*) par un certain plan si ce plan est exécutable. Nous pouvons ainsi définir les notions de but *a-réalisable*, *r-réalisable* et *s-réalisable* :

$$achievable_a(g,p) \Leftrightarrow achieves_g(p,g) \wedge feasible_a(p) \quad (6.6)$$
$$achievable_r(g,p) \Leftrightarrow achieves_g(p,g) \wedge feasible_r(p) \quad (6.7)$$
$$achievable_s(g,p) \Leftrightarrow achievable_a(g,p) \wedge achievable_r(g,p) \quad (6.8)$$

Par la suite, nous adoptons les mêmes hypothèses simplificatrices introduites dans la section 5.3.8 concernant le type de dépendance. Nous ne nous intéressons ainsi qu'aux notions concernant les actions, et utilisons respectivement les termes exécutable et réalisable comme des synonymes de a-exécutable et de a-réalisable.

Nous montrons dans la suite comment les divers choix d'un agent sont réalisés selon notre modèle. Nous avons implicitement adopté l'hypothèse que les agents choisissent d'abord un but à accomplir, ensuite un plan qui réalise ce but et, seulement après, des partenaires possibles.

6.2 Choix de buts

Dans le contexte de notre étude, un agent peut utiliser un plan comportant des actions qu'il ne sait pas entreprendre. Par conséquent, lorsqu'il doit choisir un but à atteindre à un certain moment, il doit examiner si ce but est réalisable ou non. Dans les sections suivantes, nous présentons de façon informelle cette procédure, nous proposons un modèle pour la caractériser et nous montrons comment le modèle d'agent introduit dans la section 3.4 est capable de la gérer.

6.2.1 Procédure de choix de buts

Comme nous l'avons expliqué dans la section 5.3.8, un agent considère toujours, au départ, ses propres plans. Pour choisir un but à atteindre à un instant donné,

un agent i calcule si ce but est réalisable :

$$achievable(i,g) \Leftrightarrow AUT(i,g) \vee (DEP(i,g) \wedge \exists p\ (is_p(i,p) \wedge achievable_a(g,p)) \quad (6.9)$$

Un agent infère qu'un but g est réalisable si (i) il est autonome pour ce but ou (ii) il existe un plan dont toutes les actions nécessaires peuvent être entreprises par au moins un agent appartenant à la société. L'analyse de l'équation 5.44 montre que $AUT(i,g) \Rightarrow achievable(i,g)$, car l'agent peut entreprendre lui même toutes les actions nécessaires dans le plan.

Prenons comme exemple le cas d'un chercheur, $ag2$, dans un laboratoire de recherche[1]. Supposons que ce chercheur ait deux buts à accomplir : écrire un article sur la simulation sociale ($write\_ss\_paper$) et un autre sur une approche multi-agent pour la simulation sociale ($write\_ss\_mas\_paper$). Supposons encore que ses plans pour ces buts soient respectivement :

$$\pi_1 : write\_ss\_paper() := write\_ss\_section(), process\_word().$$
$$\pi_2 : write\_ss\_paper() := write\_ss\_section(), process\_latex().$$

pour le premier but $write\_ss\_paper$ et :

$$\pi_3 : write\_ss\_mas\_paper() := write\_ss\_section(), write\_mas\_section(), process\_word().$$

pour le deuxième but $write\_ss\_mas\_paper$. Supposons encore que ce chercheur connaisse le logiciel de traitement de textes WORD mais pas le langage LaTeX, et qu'il soit capable d'écrire la section relative à la simulation sociale des deux articles, mais pas celle concernant les systèmes multi-agents.

Si l'action $write\_mas\_section$ ne peut être entreprise par aucun chercheur dans son laboratoire, en utilisant son mécanisme de raisonnement social, l'agent $ag2$ déduit les deux propositions suivantes :

$$achievable(ag2, write\_ss\_paper)$$
$$\neg achievable(ag2, write\_ss\_mas\_paper)$$

Dans notre modèle de description externe présentée dans la section 5.1, nous avons introduit une fonction $w(g)$ pour représenter l'importance d'un but. Selon une approche purement utilitaire, nous pouvons supposer qu'étant donné deux buts g et g' tels que $w(g) > w(g')$, un agent choisit d'atteindre le but g, auquel il associe une importance plus grande. Nous montrons ici qu'un choix de but fondé exclusivement sur cette notion d'importance est *insuffisant* pour caractériser des sociétés ouvertes. Il ne semble pas raisonnable qu'un agent choisisse un but sur le seul critère d'importance sans tenir compte de la possibilité de le réaliser ou

[1] Cet exemple sera détaillé dans le chapitre 10.

non[2]. Dans notre exemple, même en supposant que $w(write\_ss\_mas\_paper) > w(write\_ss\_paper)$ (par exemple parce qu'il s'agit d'une conférence plus importante), l'agent $ag2$ choisira le but $write\_ss\_paper$, car son mécanisme de raisonnement social lui a permit de détecter que l'autre but n'est pas réalisable à ce moment-là.

6.2.2 Critère de choix de buts

Soit $M^G = (i, G, achievable_{goal}, w_{goal}, \prec_{goal}, decision_{goal})$ un modèle pour caractériser le comportement d'un agent en ce qui concerne son choix de buts tel que :

- i désigne l'agent à qui ce modèle appartient ;
- G désigne l'ensemble de ses buts ;
- $achievable_{goal} : G \mapsto B$ est une fonction qui donne pour chaque but $g \in G$ une valeur booléenne pour exprimer si'il est réalisable ou non ;
- $w_{goal} : G \mapsto N$ est une fonction qui donne pour chaque but $g \in G$ une valeur entière qui exprime son importance ;
- $\prec_{goal}$ est une relation d'ordre partielle définie dans la suite ;
- $decision_{goal} : 2^G \mapsto G$ est une fonction qui à partir de l'ensemble des parties de l'ensemble de buts G choisit un but g à être poursuivi, cette fonction étant définie par la suite.

Notons $\prec_{goal}$ la relation sur G^2 suivante :

Définition 6.2.1. *Si $g, g' \in G$ alors $g' \prec_{goal} g$ ssi :*

- $achievable_{goal}(g') = faux$ *et* $achievable_{goal}(g) = vrai$ *ou*
- $achievable_{goal}(g') = achievable_{goal}(g) = vrai$ *et* $w_{goal}(g') < w_{goal}(g)$.

La relation $\prec_{goal}$ étant transitive et anti-symétrique, nous avons que le lemme suivant est valable :

Lemme 6.2.2. $(G, \prec_{goal})$ *est un ensemble partiellement ordonné.*

Cet ordre est représenté sous forme de diagramme dans la figure 6.1.

[2]En effet, comme nous l'avons décrit dans la section 4.1.2, selon l'approche utilitaire tous les buts d'un agent sont considérés comme étant toujours réalisables.

Désignons par w_{max} la valeur désignant la plus grande importance de tous les buts réalisables d'un agent. Notons que cette valeur ne correspond pas nécessairement au but plus important, car celui-ci peut ne pas être réalisable. Soit G_{max} l'ensemble de buts pour lesquels l'importance est w_{max} et soit $G^c_{max} = G \cap \overline{G_{max}}$:

$$\forall g \in G_{max} \quad \nexists g' \in G^c_{max}, g' \neq g \;\; g \prec_{goal} g'$$

Maintenant, nous pouvons définir la fonction de choix de buts $decision_{goal}$ de la façon suivante :

$$decision_{goal}(G) = \begin{cases} aucun & \text{si } \forall g \in G \; achievable_{goal}(g) = faux \\ random(G_{max}) & \text{sinon} \end{cases}$$

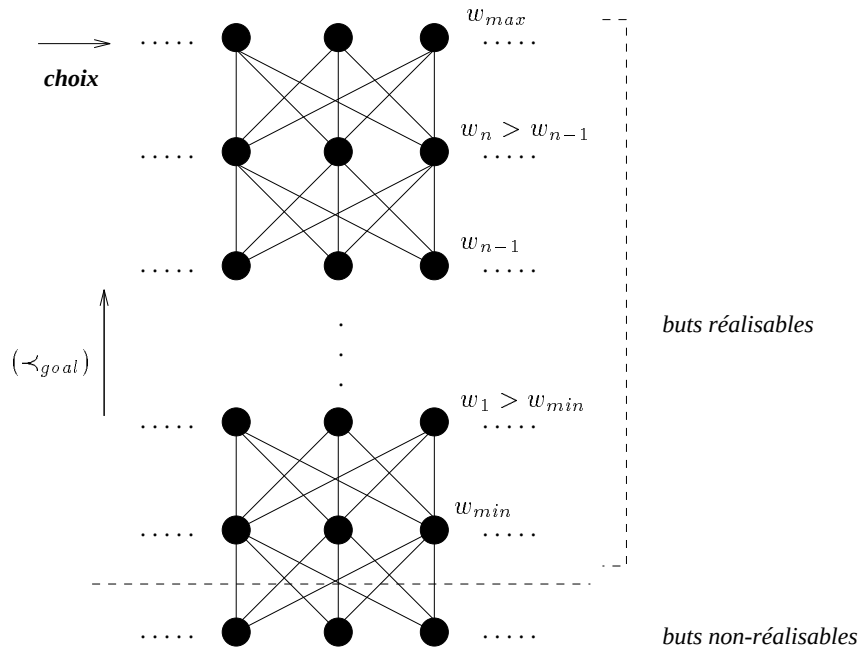


Figure 6.1 : Critère de choix de buts

Comme nous n'imposons pas que les buts doivent avoir nécessairement une importance différente, le choix d'un parmi G_{max} est fait de façon aléatoire par la fonction $random(G_{max})$. Dans le cas où tous les buts sont non-réalisables, l'agent doit attendre qu'un autre agent qui puisse rendre réalisable au moins un de ses buts entre dans la société.

6.2.3 Gestion de choix de buts par le modèle d'agent

En instanciant le critère présenté ci-dessus à notre modèle d'agent, nous avons :

- G est composé de tous les buts $g_i \in G(i)$ appartenant à sa propre entrée de description externe $ED(i)$ (équation 5.3) ;

- la fonction $achievable_{goal}(g) = vrai$ ssi le prédicat $achievable(i, g)$ (équation 6.9) est vrai ;

- la fonction w_{goal} corresponde à la composante $w(g)$ de l'équation 5.4.

La gestion de choix de buts par le modèle d'agent est présentée dans la figure 6.2.

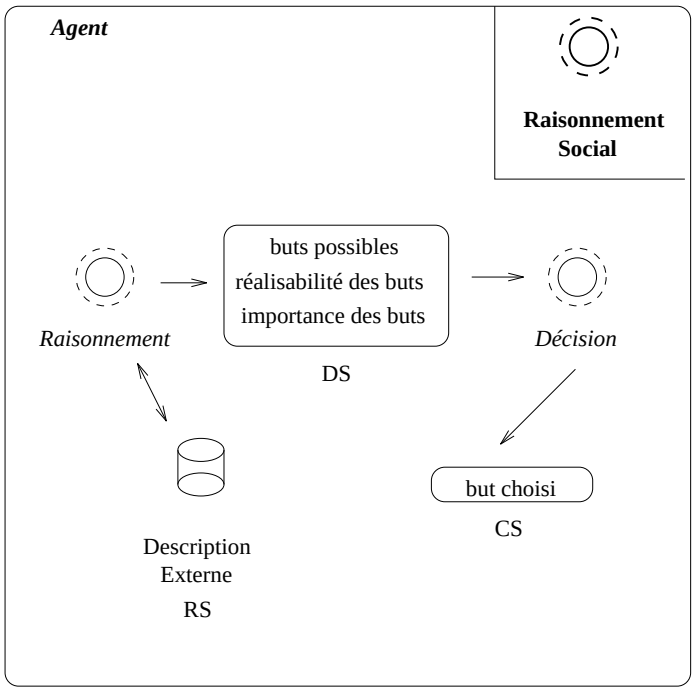


Figure 6.2 : Gestion de choix de buts par le modèle d'agent

A partir de la description externe (RS), le mécanisme de raisonnement d'un agent i calcule d'abord pour chaque but g à atteindre sa réalisabilité, que nous avons exprimé par le prédicat $achievable(i, g)$ dans la section 6.2. A la fin de ce calcul, il envoie la liste de buts possibles au mécanisme de décision (DS). Celui-ci décide le but qui doit être poursuivi en utilisant le critère $\prec_{goal}$ défini dans la section précédente (CS).

6.2.4 A propos de la genèse de buts

Dans cette thèse, nous ne nous occupons pas de la problématique de la genèse endogène de buts[3]. Des buts peuvent être générés par des désirs [RG95]. Néanmoins, nous croyons que notre modèle est suffisamment riche pour permettre de prendre en compte une génération de tel type, même si nous ne nous y intéressons pas dans le contexte de notre étude. Nous pouvons la représenter par la flêche qui part du mécanisme de raisonnement vers la description externe. Lorsqu'un but est généré, l'agent met à jour l'entrée de la description externe relative à soi-même. Evidemment, comme nous le montrons dans le chapitre suivant, ce fait aura comme conséquence dans un premier temps que les autres agents aurons une représentation incomplète de ce premier. A notre avis, ce fait est assez normal et plutôt souhaitable : *nous ne croyons pas que dans une société ouverte tous les agents puissent avoir des informations correctes et complètes des autres à tout moment.* Le problème plus évident est que pour maintenir une telle cohérence globale de représentation, il faudrait qu'un agent envoie à tous les autres un message toutes les fois qu'il génère un nouveau but[4]. Néanmoins, lorsque cette information devient importante, une mise à jour locale doit être entreprise, par exemple lorsque les destinataires d'une proposition de formation de coalition ne sont pas au courant que le proposant possède le but que la coalition doit atteindre.

6.3 Choix de plans

Une fois un but sélectionné, un agent doit choisir un plan dont l'exécution réalise ce but. Un agent peut avoir plusieurs plans pour atteindre un même but. Selon les mêmes arguments présentés dans la section précédente, avant de choisir un plan un agent doit s'assurer que celui-ci est exécutable. Dans les sections suivantes, nous présentons informellement cette procédure, nous proposons un modèle pour la caractériser et nous montrons comment le modèle d'agent introduit dans la section 3.4 est capable de la gérer.

6.3.1 Procédure de choix de plans

Comme nous l'avons expliqué dans la section 5.3.8, un agent utilise d'abord son propre ensemble de plans en cours de son raisonnement. Ainsi, le calcul de l'exécutabilité d'un plan est fait de la façon suivante :

$$feasible(i,p) \quad \Leftrightarrow \quad is_p(i,p) \wedge feasible_a(p) \tag{6.10}$$

En reprenant l'exemple de la section précédente, supposons que l'agent $ag2$ ait choisi de poursuivre le but $write\_ss\_paper$, car c'est le seul but réalisable. Pour

[3]Par contre, la genèse exogène a été abordée lorsque nous avons traité la problématique de l'adoption de buts, dans le chapitre 5.

[4]Cette explication peut être étendue aussi en ce qui concerne la planification en ligne, comme nous le montrons dans la section suivante.

l'atteindre, il peut se servir de deux plans. Supposons que dans son laboratoire aucun chercheur ne connaisse le langage LaTeX. En désignant respectivement les plans présentés dans la section précédente par π_1 et π_2, l'agent $ag2$ déduit les deux propositions suivantes :

$$feasible(ag2, \pi_1)$$
$$\neg feasible(ag2, \pi_2)$$

Dans le modèle de description externe présenté dans la section 5.1, nous avons introduit des fonctions $c(a)$ et $c(r)$ pour représenter respectivement le coût d'une action et d'une ressource. En première approximation, nous pouvons supposer que le coût associé à un plan est la somme des coûts de toutes les actions et ressources nécessaires à son exécution :

$$c(p) \triangleq \sum_{a_k \in A_p} c(a_k) + \sum_{r_k \in R_p} c(r_k) \tag{6.11}$$

où $A_p = \{a_k \mid uses_a(p, a_k)\}$ et $R_p = \{r_k \mid uses_r(p, r_k)\}$. Selon une approche purement utilitaire, nous pouvons aussi supposer qu'étant donné deux plans p et p' tels que $c(p) < c(p')$ pour atteindre un même but, un agent choisisse le plan p, car il est moins coûteux. Comme dans la section précédente, un choix de plan fondé exclusivement sur cette notion de coût est lui aussi *insuffisant* pour caractériser des sociétés ouvertes. Il ne semble pas raisonnable qu'un agent choisisse un plan moins coûteux, mais non-exécutable. Dans notre exemple, même en supposant que $c(\pi_1) > c(\pi_2)$ (par exemple parce que écrire un texte en LaTeX prend moins de temps), l'agent $ag2$ choisira le plan π_1, car son mécanisme de raisonnement social lui a permit de détecter que l'autre plan n'est pas exécutable à ce moment-là.

6.3.2 Critère de choix de plans

Soit $M^P = (i, g, P, feasible_{plan}, c_{plan}, \prec_{plan}, decision_{plan})$ un modèle pour caractériser le comportement d'un agent en ce qui concerne son choix de plans tel que :

- i désigne l'agent à qui ce modèle appartient ;
- g désigne le but étant poursuivi ;
- P désigne l'ensemble de ses plans qui atteignent le but g ;
- $feasible_{plan} : P \mapsto B$ est une fonction qui donne pour chaque plan $p \in P$ une valeur booléenne pour exprimer s'il est exécutable ou non ;
- $c_{plan} : P \mapsto N$ est une fonction qui donne pour chaque plan $p \in P$ une valeur entière qui exprime son coût ;
- $\prec_{plan}$ est une relation d'ordre partielle définie dans la suite ;

- $decision_{plan} : 2^P \mapsto P$ est une fonction qui à partir de l'ensemble des parties de l'ensemble de plans P choisit un plan p à être exécuté, cette fonction étant définie dans la suite.

Notons $\prec_{plan}$ la relation sur P^2 suivante :

Définition 6.3.1. *Si $p, p' \in P$ alors $p' \prec_{plan} p$ ssi :*

- $feasible_{plan}(p') = faux$ *et* $feasible_{plan}(p) = vrai$ *ou*
- $feasible_{plan}(p') = feasible_{plan}(p) = vrai$ *et* $c_{plan}(p') < c_{plan}(p)$.

La relation $\prec_{plan}$ étant visiblement transitive et anti-symétrique, nous avons que le lemme suivant est valable :

Lemme 6.3.2. $(P, \prec_{plan})$ *est un ensemble partiellement ordonné.*

Cet ordre est représenté sous forme de diagramme dans la figure 6.3.

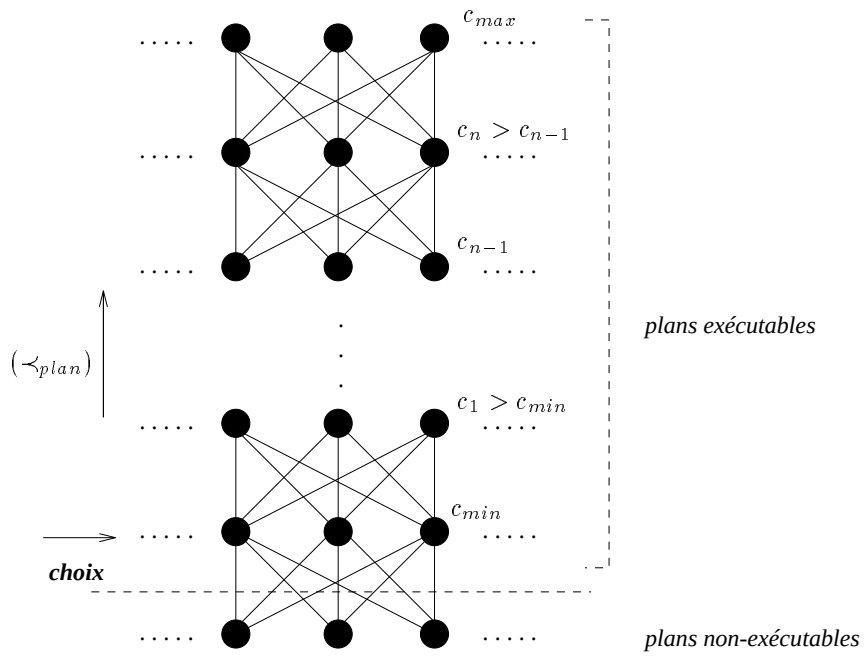


Figure 6.3 : Critère de choix de plans

Désignons par c_{min} la valeur désignant le plus petit coût de tous les plans exécutables d'un agent pour le but g. Notons que cette valeur ne correspond pas nécessairement au plan moins coûteux, car celui-ci peut ne pas être exécutable. Soit P_{min} l'ensemble de plans pour lesquels le coût est c_{min} et soit $P^c_{min} = P \cap \overline{P_{min}}$:

$$\forall p \in P_{min} \quad \nexists p' \in P^c_{min}, p' \neq p \;\; p' \prec_{plan} p \wedge feasible_{plan}(p') = vrai$$

Maintenant, nous pouvons définir la fonction de choix de plans $decision_{plan}$ de la façon suivante :

$$decision_{plan}(P) = random(P_{min})$$

Comme nous n'imposons pas que que les plans doivent avoir nécessairement un coût différent, le choix d'un parmi P_{min} est fait de façon aléatoire par la fonction $random(P_{min})$. Notons que par notre définition de but réalisable (équation 6.9), nous garantissons qu'il existe au moins un plan exécutable qui l'atteint.

6.3.3 Gestion de choix de plans par le modèle d'agent

En instanciant le critère présenté ci-dessus à notre modèle d'agent, nous avons :

- le but poursuivi g fait partie de l'état d'engagement CS généré dans l'étape précédente ;
- P est composé de tous les plans $p_i \in P(i)$ appartenant à sa propre entrée de description externe $ED(i)$ (équation 5.9) tels que le prédicat $achieves(p, g)$ défini dans la section 5.3.1 est vrai ;
- la fonction $feasible_{plan}(p) = vrai$ ssi le prédicat $feasible(i, p)$ (équation 6.10) est vrai ;
- la fonction c_{plan} est calculée selon l'équation 6.11, en utilisant les coûts d'actions et de ressources définies dans les équations 5.5 et 5.7.

La gestion de choix de plans par le modèle d'agent est présentée dans la figure 6.4. A partir du but choisi dans l'étape précédente (CS) et de la description externe (RS), le mécanisme de raisonnement d'un agent i calcule pour chaque plan p qui atteint le but g son exécutabilité, que nous avons exprimé par le prédicat $feasible(i, p)$ dans la section 6.3. A la fin de ce calcul, il envoie la liste de plans possibles au mécanisme de décision (DS). Celui-ci décide le plan à être adopté en utilisant le critère $\prec_{plan}$ défini dans la section précédente (CS). Notons que selon les prédicats 6.9 et 6.10, un agent peut bien en avoir des plans non-exécutables pour un but réalisable.

6.3.4 A propos de la planification

Comme nous l'avons dit dans la section 3.4, nous n'abordons pas dans cette thèse la problématique de la planification. De cette façon, nous utilisons la situation de but NP (équation 5.43) pour désigner la situation où un agent a un but qu'il ne sait pas comment atteindre. Si un agent est dans cette situation, il sera toujours réceptif à une proposition de coopération, puisque cette dernière lui permet de l'atteindre.

Néanmoins, nous croyons que notre modèle est suffisamment riche pour permettre de prendre en compte une planification en ligne, même si nous ne nous y

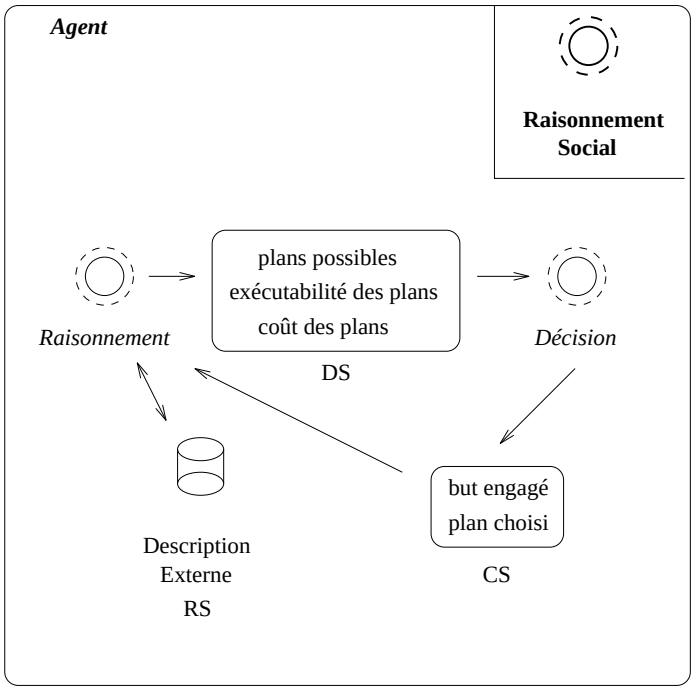


Figure 6.4 : Gestion de choix de plans par le modèle d'agent

intéressons pas dans le contexte de notre étude. Supposons que dans l'étape précédente le mécanisme de décision a choisi un but g pour lequel l'agent i n'a pas de plans pré-établis pour l'atteindre, c'est-à-dire, dont la situation de but est *NP(i,g)*. Comme dans le cas de la génération endogène de buts, nous pouvons représenter cette phase de planification par la flèche qui part du mécanisme de raisonnement vers la description externe. Une fois de plus, lorsqu'un plan est généré, l'agent met à jour l'entrée de la description externe relative à soi-même. Les remarques faites dans la section précédente à propos d'éventuelles incomplétudes au niveau de la société d'agents sont également valables ici.

6.4 Discussion

Dans ce chapitre, nous avons montré comment un mécanisme de raisonnement social peut être exploité par un agent pour choisir un but à poursuivre ou un plan à exécuter, en lui permettant de s'adapter dans un contexte d'un SMA ouvert.

A notre connaissance, aucune méthode n'a encore proposé d'étudier l'influence de la réalisabilité des buts et de l'exécutabilité des plans dans les choix possibles d'un agent, qui est placé dans une société ouverte. En ce qui concerne cet aspect, nous croyons que nos travaux sont assez novateurs.

Un autre résultat intéressant, selon la perspective SS, concerne le *pouvoir social*

des agents. Au fur et à mesure qu'une société évolue, le nombre d'agents dont un agent dépend ou qui dépendent de lui change. Ainsi, le pouvoir social d'un agent change lui aussi. Nous allons discuter ce point plus en détail dans le chapitre 10, en étendant l'exemple du laboratoire de recherche présenté dans ce chapitre.

Une fois choisi un but et un plan, un agent peut calculer sa situation de but pour évaluer s'il est capable d'exécuter le plan choisi tout seul. Si le résultat est *AUT*, alors il l'exécute. Par contre, si le résultat est *DEP*, il faut qu'il propose aux autres la formation d'une coalition pour atteindre son but. D'ailleurs, dans le cas où il existe plus d'un partenaire possible qui peut entreprendre une certaine action dont il en a besoin, il doit être capable d'évaluer lequel est le plus susceptible d'accepter d'en faire partie. Nous montrons dans le chapitre suivant comment les situations de dépendance peuvent être utilisées à cet égard.

Chapitre 7

Effet sur la formation de coalitions

Dans ce chapitre, nous analysons l'effet du mécanisme de raisonnement social sur la *formation de coalitions entre agents*. En nous plaçant dans un contexte d'agents non-bienveillants, nous montrons qu'un agent peut utiliser les situations de dépendance inférées par ce mécanisme pour choisir les possibles partenaires à qui envoyer ou de qui accepter une proposition de formation de coalition lorsqu'il est incapable d'atteindre son but tout seul. En outre, nous analysons l'évolution de ces choix, en démontrant que notre mécanisme de raisonnement social permet à un agent de les changer dans un contexte d'une société ouverte. Nous avons initialement présenté les critères de choix de partenaires dans [SD94a] et l'évolution des choix dans [SD95b].

Ce chapitre est organisé de la façon suivante. Nous présentons la problématique de choix de partenaires dans la section 7.1, où nous proposons deux critères fondés respectivement sur la *source des plans* et la *nature de la dépendance*. Nous proposons ensuite qu'un agent utilise la combinaison de ces critères pour choisir ses partenaires. Dans la section 7.2, nous discutons les éventuelles différences entre le choix et l'acceptation de partenaires. Pour chacun de ces choix, nous décrivons la procédure qui doit être réalisée, un critère formel qui l'exprime et sa gestion par le modèle d'agent. Nous présentons un exemple détaillé de l'évolution de ces choix dans la section 7.3. Enfin, nous terminons ce chapitre en présentant dans la section 7.4 une discussion sur les résultats obtenus et une comparaison avec d'autres approches.

7.1 Choix de partenaires

Un but et un plan à exécuter ayant été sélectionnés, un agent doit aussi choisir les partenaires à qui proposer de former une coalition lorsqu'il est incapable d'exécuter le plan tout seul. Imaginons que dans l'exemple du laboratoire, il existe maintenant deux chercheurs connaissant le domaine des SMA, c'est-à-dire,

l'action *write_mas_section* est devenue disponible. Pour cette raison l'agent $ag2$ veut maintenant choisir le but $write\_ss\_mas\_paper()$. La question qui s'impose maintenant est la suivante: *A quel agent doit-elle envoyer une proposition de coopération ou d'échange ?*

Comme nous n'adoptons pas dans ce travail l'hypothèse de bienveillance entre des agents, nous proposons d'utiliser les situations de dépendance introduites dans la section 5.3.10 comme un *critère de décision* lorsqu'un agent doit choisir ses partenaires. Autrement dit, si un agent i a besoin d'une action a pour atteindre un but g, et s'il existe un ensemble d'agents $j, k, \ldots$ qui peuvent entreprendre cette action, l'agent peut calculer pour le but g sa situation de dépendance envers chacun des agents pouvant entreprendre cette action et ainsi choisir de s'adresser à celui dont il suppose que la réponse sera la plus favorable.

Dans les sections suivantes, nous présentons de façon informelle cette procédure, nous proposons un modèle pour la caractériser et nous montrons comment le modèle d'agent introduit dans la section 3.4 est capable de la gérer.

7.1.1 Procédure de choix de partenaires

Dans le reste de ce chapitre, nous désignerons par *proposant* un agent qui envoie à un autre une proposition de formation de coalition et par *destinataire* l'agent qui reçoit cette proposition. L'intention du proposant est que le destinataire *adopte son but* et entreprenne l'action dont il a besoin. Le destinataire peut néanmoins rejeter la proposition, dans le cas où il préfère participer à une autre coalition avec d'autres agents.

Pour représenter formellement notre problème, soit $DSIT$ un ensemble dont les éléments sont des situations de dépendance possibles entre deux agents pour un but particulier :

$$DSIT = \{IND, UD, LBRD, LBMD, MBRD, MBMD\} \tag{7.1}$$

Nous définissons deux relations $\prec_{source}$ et $\prec_{nature}$ sur $DSIT^2$, pour exprimer formellement nos critères : la première prend en compte la source des plans et la deuxième la nature de la dépendance.

7.1.2 Critère de la source de plans

Nous appelons *source* le critère qui distingue les situations de dépendance localement reconnues de celles qui sont mutuellement reconnues. En rappelant les définitions introduites dans la section 5.3.10, le proposant déduit une dépendance localement reconnue s'il n'utilise que ses propres plans pour l'inférer, tandis qu'il infère une dépendance mutuellement reconnue s'il utilise à la fois ses plans et ceux qu'il croit appartenir au destinataire.

Evidement, la situation de dépendance *IND* est inutile, car il est évident qu'un proposant ne s'adressera jamais à un agent incapable d'entreprendre l'action dont il a besoin.

Nous considérons que *si le proposant a inféré une dépendance mutuellement reconnue, il aura plus de chances d'obtenir l'action qu'il désire.* En supposant que les deux agents ont la même représentation l'un de l'autre[1], nous sommes dans la situation où les deux agents infèrent la même situation de dépendance l'un envers l'autre et par conséquent sont tous les deux *conscients* de cette situation. Concernant le flux de communication globale, les interactions seront plus simples, car le proposant n'a pas besoin de convaincre le destinataire de leur situation de dépendance. Si nous associons une fonction de coût à la communication, ces situations ($MBRD$ et $MBMD$) coûteront moins au proposant, et seront donc préférables aux autres.

Notons $\prec'_{source}$ la relation sur $DSIT^2$ suivante :

Définition 7.1.1. *Si* $a, b \in DSIT$ *alors* $a \prec'_{source} b$ *ssi :*

- $a \in \{IND\}$ *et* $b \in \{UD, LBMD, LBRD, MBMD, MBRD\}$ *ou*
- $a \in \{UD, LBRD, LBMD\}$ *et* $b \in \{MBRD, MBMD\}$.

Soit $\prec_{source}$ la fermeture transitive de $\prec'_{source}$. La preuve du lemme suivant est triviale, puisque la relation $\prec'_{source}$ est anti-symétrique :

Lemme 7.1.2. $(DSIT, \prec_{source})$ *est un ensemble partiellement ordonné.*

Ce critère est représenté sous forme de diagramme dans la figure 7.1.

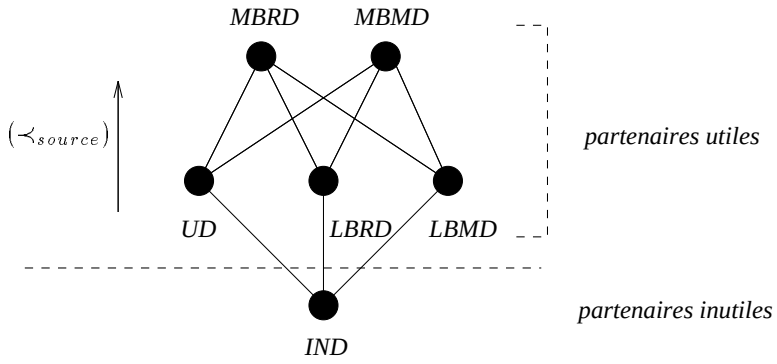


Figure 7.1 : Critère de la source de plans

7.1.3 Critère de la nature de la dépendance

Nous appelons *nature* le critère qui distingue les situations de dépendance selon ce qui est offert par le proposant au destinataire en échange. Nous considérons que

[1] Nous appellerons cette situation *hypothèse de compatibilité de description externe*, et la détaillerons dans le chapitre 8.

si le proposant peut offrir quelque chose en échange d'une action qu'il demande, il a plus de chances d'obtenir l'action qu'il désire.

Une fois de plus, la situation de dépendance *IND* est inutile. De façon évidente, la deuxième situation la moins avantageuse est une *UD*, car le proposant n'a rien à offrir.

En comparant une dépendance mutuelle et une réciproque, nous considérons aussi qu'*une dépendance mutuelle est une meilleure option pour le proposant.* L'explication de cette hypothèse est que le destinataire, en ayant le même but à accomplir, ne doit pas craindre l'absence de réciprocité de la part du proposant. Dans le cas d'une *LBRD* ou d'une *MBRD*, un des agents doit nécessairement d'abord adopter le but de l'autre. Il peut alors se poser éventuellement la question de savoir si l'autre va effectivement accomplir son engagement futur. Cette question est rarement abordée dans le cadre d'agents bienveillants. Notre modèle a ainsi l'avantage de ne présupposer aucun comportement pré-défini des agents.

Notons $\prec'_{nature}$ la relation sur $DSIT^2$ suivante :

Définition 7.1.3. *Si* $a, b \in DSIT$ *alors* $a \prec'_{nature} b$ *ssi :*

- $a \in \{IND\}$ *et* $b \in \{UD, LBMD, LBRD, MBMD, MBRD\}$ *ou*
- $a \in \{UD\}$ *et* $b \in \{LBRD, MBRD\}$ *ou*
- $a \in \{LBRD, MBRD\}$ *et* $b \in \{LBMD, MBMD\}$.

Soit $\prec_{nature}$ la fermeture transitive de $\prec'_{nature}$. La preuve du lemme suivant est évidente, puisque la relation $\prec'_{nature}$ est anti-symétrique :

Lemme 7.1.4. $(DSIT, \prec_{nature})$ *est un ensemble partiellement ordonné.*

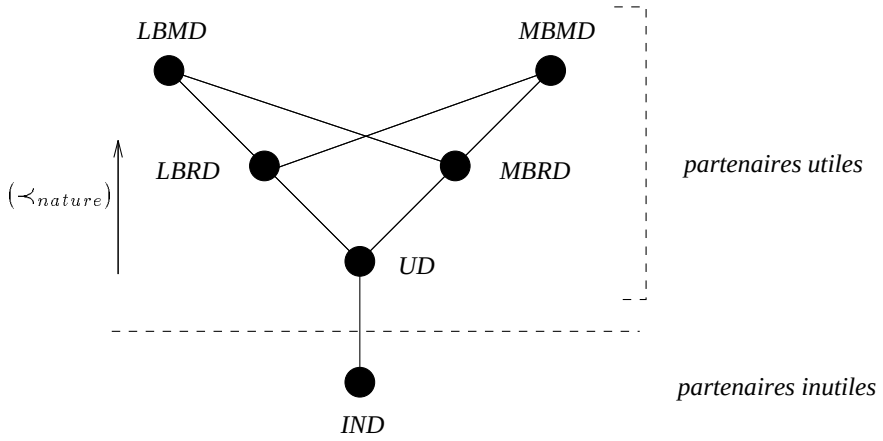


Figure 7.2 : Critère de la nature de la dépendance

Ce critère est représenté sous forme de diagramme dans la figure 7.2.

7.1.4 Critère de choix de partenaires

Nous allons maintenant combiner les deux critères précédents, afin de permettre à un agent de choisir le meilleur partenaire possible pour atteindre un but donné.

Représentons respectivement par $a \not\prec_{source} b$ et $a \not\prec_{nature} b$ le fait que la paire (a, b) n'appartient à aucune des relations précédentes.

Soit $\prec_{partner}$ une relation sur $DSIT^2$, qui combine les deux relations précédentes de la façon suivante :

Définition 7.1.5. *Si $a, b \in DSIT$ alors $a \prec_{partner} b$ ssi :*

- $a \prec_{source} b$ *et* $a \prec_{nature} b$ *ou*
- $a \prec_{source} b$ *et* $a \not\prec_{nature} b$ *et* $b \not\prec_{nature} a$ *ou*
- $a \prec_{nature} b$ *et* $a \not\prec_{source} b$ *et* $b \not\prec_{source} a$.

La preuve des lemmes suivants est évidente :

Lemme 7.1.6. *$(DSIT, \prec_{partner})$ est un ensemble partiellement ordonné.*

Lemme 7.1.7. *$(DSIT, \prec_{partner})$ est un treillis, contenant un élément minimum (IND), deux éléments non comparables (LBMD et MBRD) et un élément maximum (MBMD).*

Cette combinaison de critères, présenté sous forme de treillis dans la figure 7.3, correspond à l'ordre de préférences des partenaires qu'un agent utilisera pour décider à qui envoyer une proposition de formation de coalition. Selon cet ordre, un agent préférera toujours s'adresser à celui envers lequel il aura inféré une *MBMD*, qui correspond à la situation de dépendance la plus avantageuse. En revanche, il va toujours éviter d'interagir avec ceux envers lesquels il aura inféré une *UD*, qui est la situation la moins avantageuse.

Etant défini le critère ci-dessus, nous pouvons maintenant proposer un modèle pour caractériser le comportement d'un agent en ce qui concerne son choix de partenaires.

Soit $GSIT$ un ensemble dont les éléments sont des situations de but tel que nous les avons définies dans la section 5.3.9 :

$$GSIT = \{NG, NP, AUT, DEP\} \tag{7.2}$$

Soit $M^C = (i, g, p, G, P, A, GSIT, DSIT, goalsit, depsit, \prec_{partner}, decide_{patners})$ tel que :

- i désigne l'agent à qui ce modèle appartient ;
- g désigne le but étant poursuivi ;

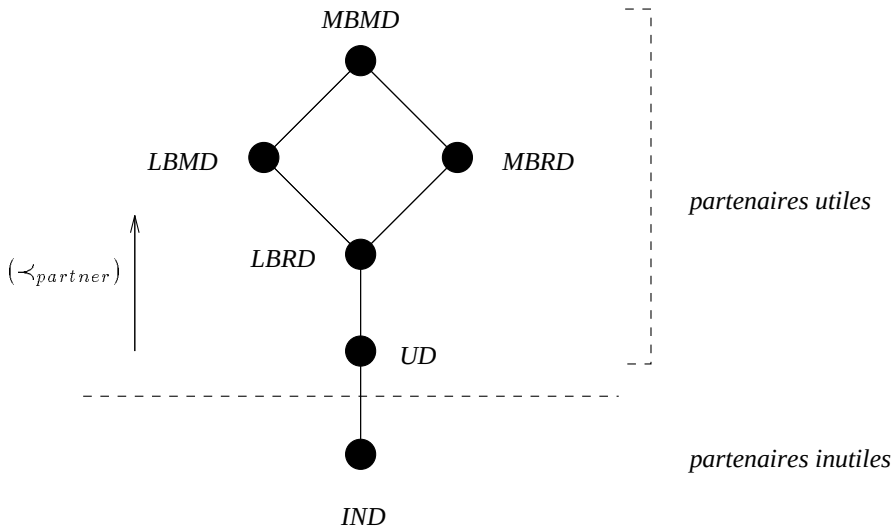


Figure 7.3 : Critère de choix de partenaires

- p désigne le plan étant choisi pour atteindre g ;
- G désigne l'ensemble de buts de l'agent ;
- P désigne l'ensemble de plans de l'agent pour le but g ;
- A est un ensemble d'agents ;
- $GSIT$ désigne l'ensemble de situations de buts (équation 7.2) ;
- $DSIT$ désigne l'ensemble de situations de dépendance (équation 7.1) ;
- $goalsit : G \times 2^P \mapsto GSIT$ est une fonction qui donne pour chaque but g et pour l'ensemble de plans $\{p\}$[2] la situation de but de l'agent i, comme nous l'avons introduit dans la section 5.3.9 ;
- $depsit : G \times 2^P \times A \mapsto DSIT$ est une fonction qui donne pour chaque but g et pour l'ensemble de plans $\{p\}$ la situation de dépendance de l'agent i envers un agent $j \in A$, comme nous l'avons introduit dans la section 5.3.10 ;
- $\prec_{partner}$ est la relation d'ordre partielle définie précédemment ;

[2]Ici il y a une petite astuce technique, car les notion de situation de but et de situations de dépendance sont définies à partir d'un *ensemble de plans*. Donc, une fois un plan choisi, nous pouvons bien le représenter comme un ensemble n'ayant que lui seul comme élément. Nous proposons quelques améliorations concernant cet aspect dans nos perspectives présentées dans la section 12.2.

- $decide_{partner}$: $G \times 2^P \times 2^A \mapsto A$ est une fonction qui à partir de g, de l'ensemble $\{p\}$ et de l'ensemble A choisit un partenaire j auquel une proposition de formation de coalition doit être envoyée, cette fonction étant définie dans la suite.

Désignons par $dsit_{max}$ la situation de dépendance la plus avantageuse selon le critère $\prec_{partner}$ calculée par la fonction $depsit$ pour tout agent $j \in A$. Soit A_{max} l'ensemble d'agents envers lesquels i calcule une situation de dépendance du type $dsit_{max}$ et soit $A^c_{max} = A \cap \overline{A_{max}}$:

$$\forall j \in A_{max} \quad \nexists k \in A^c_{max}, k \neq j \; depsit(g, \{p\}, j) \prec_{partner} depsit(g, \{p\}, k)$$

Maintenant, nous pouvons définir la fonction $decide_{partner}$ de choix de partenaires de la façon suivante :

$$decide_{partner}(g, \{p\}, A) = \begin{cases} aucun & \text{si } goalsit(g, \{p\}) = AUT \\ random(A_{max}) & \text{sinon} \end{cases}$$

Comme l'agent i peut avoir inféré la situation de dépendance $dsit_{max}$ envers plusieurs agents, le choix d'un parmi A_{max} est fait de façon aléatoire par la fonction $random(A_{max})$. Nous voulons remarquer trois points importants : (i) nous avons supposé qu'en étant autonome (première cas de la fonction) l'agent va toujours preférer travailler tout seul. Dans nos perspectives, décrites dans la section 12.2, nous analysons ce point plus en détail ; (ii) une fois que le but g est réalisable et le plan p est exécutable, si $goalsit(g, \{p\}) = DEP$, il existe au moins un agent dans la société capable d'entreprendre chaque action que l'agent i a besoin pour atteindre le but g et (iii) selon le critère $\prec_{partner}$, les situations de dépendance $LBMD$ et $MBRD$ ne sont pas comparables, ainsi les deux pourraient ne pas être distinguées lorsque $dsit_{max}$ a comme valeur l'une ou l'autre. Nous aurions pu avoir introduit un autre critère supplémentaire pour résoudre ce choix. Par exemple, nous aurions pu envisager qu'un agent puisse garder un historique des collaborations passées et choisir le partenaire dont le nombre de collaborations réussies dans le passé est le plus grand. Néanmoins, nous croyons que pour les propos de nos travaux, un critère additionnel n'aurait comme effet que la complication desnecéssaire de notre modèle. Ainsi, nous avons choisi de ne pas différencier ces deux situations.

7.1.5 Gestion de choix de partenaires par le modèle d'agent

En instanciant le critère présenté ci-dessus à notre modèle d'agent, nous avons :

- le but poursuivi g et le plan choisi pour l'atteindre p font partie de l'état d'engagement CS généré dans les deux étapes précédentes ;
- G et P sont définis respectivement de façon analogue aux sections 6.2.3 et 6.3.3 ;

- l'ensemble A contient tous les agents appartenant à la description externe ED de i (équation 5.1).

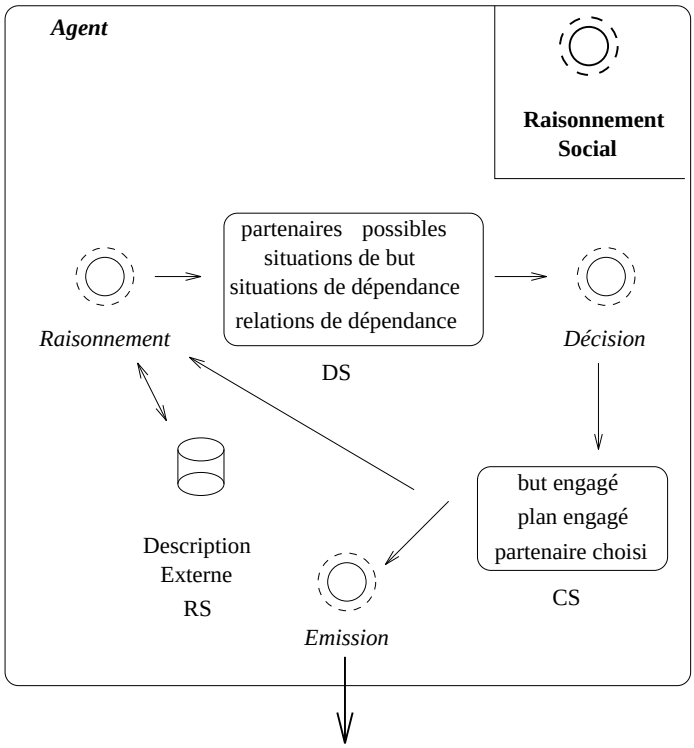


Figure 7.4 : Gestion de choix de partenaires par le modèle d'agent

La gestion de choix de partenaires par le modèle d'agent est présentée dans la figure 7.4. A partir du but et du plan choisi dans les étapes précédentes (CS) et de la description externe(RS), le mécanisme de raisonnement d'un agent calcule ses situations de but et de dépendance envers tous les partenaires possibles pour le but poursuivi g. Dans le cas où la situation de but de l'agent est $AUT(i,g)$, l'agent entreprend tout seul toutes les actions nécessaires à la réalisation du plan[3]. Par contre, si la situation de but est $DEP(i,g)$, dans le cas général un agent peut avoir besoin de plusieurs partenaires $ag_1, ag_2, \cdots, ag_n$ différents, chacun d'entre eux pouvant entreprendre une action (ou contrôlant une ressource) différente[4]. Le mécanisme de décision choisit chacun de ces partenaires $ag_i, i \in \{1, \ldots, n\}$ en utilisant le critère $\prec_{partner}$ présenté ci-dessus. Nous n'avons pas traité dans cette thèse le cas où un même partenaire peut exécuter plus d'une action dans un plan. A priori, cette situation ne pose pas de problèmes, sauf si les actions doivent être

[3]Ce cas n'est pas représenté dans la figure, car ce qui nous interesse c'est la procédure de formation de coalitions.

[4]Cette notion a été désignée comme dépendance multi-partite dans la section 4.6.1.

entreprises de façon concurrante. Par conséquent, nous croyons que notre modèle doit subir certaines adaptations pour pouvoir être appliqué dans un contexte de planification multi-agents, où on adresse souvent des actions concurrentes.

7.1.6 A propos du principe de bienveillance

Notons que dans la figure 7.4, nous avons mis aussi les *relations de dépendance* comme un critère de choix[5]. A première vue, il peut sembler peu naturel que les relations de dépendance puissent être utilisées comme un critère de choix de partenaires, puisque nous avons insisté dès le début sur le fait que dans notre contexte d'étude les agents ne sont pas nécessairement bienveillants. La réponse à cette question est simple : *même si notre modèle a été conçu pour être utilisé dans le contexte d'une société ouverte où les agents ne sont pas a priori bienveillants, ce modèle est suffisamment général pour pouvoir aussi être utilisé dans un contexte où cette hypothèse de bienveillance est valable.* Dans ce cas, l'utilisation des relations de dépendance pour le choix de partenaires est suffisante, et les situations de dépendance n'ont pas de sens. A la rigueur, dans ce contexte, la composition de la description externe peut aussi être simplifiée : les agents n'ont pas besoin de représenter ni les buts ni les plans des autres, puisque ceux-ci ne sont utilisés que pour le calcul des situations de dépendance. Dans le cas général, où les agents ne sont pas bienveillants, ce choix est basé sur le critère $\prec_{partner}$.

Une fois le(s) partenaire(s) choisi(s), l'agent leur envoie une proposition de formation de coalition. Ces derniers doivent pour leur part décider si'ils acceptent d'en faire partie ou pas.

7.2 Acceptation de partenaires

En recevant une proposition de coalition, un agent doit décider s'il doit en faire partie ou non. Dans le cas général, cette procédure est beaucoup plus compliquée que celles que nous avons décrit jusqu'à présent :

- nous ne pouvons pas garantir que les informations relatives au destinataire que le proposant possède sont correctes. Ainsi, une détection de possibles inconsistances doit être faite, comme nous le montrons dans le chapitre 8 ;

- même si dans notre contexte nous ne traitons pas directement la problématique de la planification, rien ne peut garantir que les plans choisis par deux agents pour atteindre un même but soient les mêmes, et ainsi une procédure de négociation de plans peut être nécessaire, comme nous l'avons montré lorsque nous avons introduit le modèle de résolution coopérative de problèmes dans la section 3.2. D'ailleurs, cet aspect a un effet plus négatif que

[5]Pour calculer ses situations de buts et ses situations de dépendance, un agent doit nécessairement avoir calculé d'abord ses relations de dépendance, selon le modèle présenté dans le chapitre 5.

l'on pourrait imaginer a priori : il devient possible qu'en choissant un autre plan, l'agent destinataire ne croit pas qu'il soit convenable de faire partie d'une coalition avec le proposant ;

- dans le cadre de la théorie de la dépendance, il n'existe pas jusqu'à présent des indications précises sur cette procédure.

Ainsi, nous croyons que cette tâche mérite beaucoup plus de refléxion que nous avons pu lui consacrer. Ceci-dit, nous proposons dans la suite un schéma très restrictif pour l'accomplir. Notre objectif majeur étant de montrer l'importance du mécanisme de raisonnement social au sein d'un agent, nous croyons que ce schéma peut être utile pour illustrer quelques aspects que nous trouvons importants.

7.2.1 Procédure d'acceptation de partenaires

Nous supposons qu'en recevant une proposition de coalition pour un certain but, un agent aura le comportement suivant :

1. tout d'abord, en nous rappelant de la discussion faite dans la section 5.1.2 à propos des sources d'information de la description externe, nous ne supposons pas que les agents ont toujours une représentation correcte et complète les uns sur les autres. Ainsi, un mécanisme de révision de croyances doit être activé lorsqu'un message proposant la formation d'une coalition arrive. Nous analysons ce mécanisme avec plus de détails dans le chapitre 8. Dans le cas où le destinataire détecte que le proposant a des croyances fausses relatives à lui, il refuse d'en faire partie et lui envoie l'information qui a genéré l'inconsistance. De cette façon, le proposant peut mettre à jour sa représentation du destinataire ;

2. le destinataire peut aussi détecter qu'il avait des croyances fausses ou incomplètes relatives au proposant. Dans ce cas, il doit aussi les réviser et les mettre à jour, cet aspect étant aussi abordé dans le chapitre 8 ;

3. la situation de but est alors calculée. Si celle-ci est NG ou AUT, l'agent refuse d'en faire partie ;

4. si, par contre, la situation de but est NP, l'agent accepte d'en faire partie, car de cette façon son but devient réalisable ;

5. finalement, si la situation de but est DEP, l'agent calcule sa situation de dépendance envers tous les autres agents pour le but en question, en utilisant un plan p choisi selon la procédure décrite dans la section 6.3. Une fois ce calcul accompli, il vérifie si l'agent proposant appartient à l'ensemble d'agents qu'il choisirait en supposant que la proposition de coalition partait de lui. Si c'est le cas, alors il accepte d'en faire partie, sinon il la refuse.

7.2.2 Critère d'acceptation de partenaires

Dans cette section, nous donnons une description plus formelle de la procédure décrite ci-dessus. Nous proposons une formalisation beaucoup plus simple que les précédentes, due à la difficulté de la tâche.

Soit $M^A = (i, j, g, goalsit, depsit, \prec_{partner}, accept_{partner})$ un modèle pour caractériser le comportement d'un agent en ce qui concerne son acceptation de partenaires tel que :

- i désigne l'agent à qui ce modèle appartient ;
- j désigne l'agent qui a envoyé la proposition de coalition ;
- g désigne le but pour lequel la coalition a été proposée. Dans le cas d'un échange social, ce but se réfère à celui qui intéresse i ;
- $goalsit$ et $depsit$ sont les fonctions définies dans la section 7.1.4 ;
- $goalsit$ et $depsit$ sont les fonctions définies dans la section 7.1.4 ;
- $\prec_{partner}$ est la relation d'ordre partielle aussi définie dans la section 7.1.4 ;
- $accept_{partner} : G \times 2^A \mapsto B$ est une fonction qui à partir de g et de l'ensemble de parties de A donne une valeur booléenne qui exprime l'acceptation de faire partie de la coalition, cette fonction étant définie dans la suite.

Nous supposons aussi que les diverses fonctions définies dans les modèles M^G, M^P et M^C soient définies dans M^A, nous ne les répétons pas par une question de concision. Ainsi, nous considérons que p est le plan choisi selon la procédure décrite dans la section 6.3 et que A_{max} soit l'ensemble d'agents auxquels i proposerait une coalition, comme nous l'avons défini dans la section 7.1.

La fonction d'acceptation de partenaires $accept_{partner}$ peut être décrite de la façon suivante :

$$accept_{partner}(g, j) = \begin{cases} non & \text{si } j \text{ a une croyance fausse relative aux actions que } i \text{ peut entreprendre}^6 \\ non & \text{si } goalsit(g, P) = NG \\ non & \text{si } goalsit(g, P) = AUT \\ oui & \text{si } goalsit(g, P) = NP \\ oui & \text{si } goalsit(g, P) = DEP \wedge j \in A_{max} \\ non & \text{si } goalsit(g, P) = DEP \wedge j \notin A_{max} \end{cases}$$

[6] Nous analysons cet aspect dans le chapitre 8.

Bien que nous croyons qu'il y a encore beaucoup à faire en ce qui concerne l'étude de cette procédure, nous croyons néanmoins que la fonction proposée ci-dessus est un modèle raisonnable pour la caractériser.

7.2.3 Gestion d'acceptation de partenaires par le modèle d'agent

Puisque le critère présenté dans la section précédente est basé sur les mêmes éléments que nous avons utilisés pour le choix de partenaires (situations de but, situations de dépendance), nous ne répétons pas ici sa gestion par le modèle d'agent. Nous voulons mettre en relief le fait que le calcul de choix de partenaires tel que nous l'avons décrit dans la section 7.1 est réalisé de la même façon, sauf que le but maintenant n'est pas généré par l'agent lui-même, mais par le proposant. Une révision de croyances y est activée aussi avant le calcul, comme nous le montrons dans le chapitre suivant.

7.3 Evolution des choix de partenaires

Dans [SD94a], nous présentons un exemple d'utilisation de cet ordre de préférences de partenaires dans un cadre *statique*, c'est-à-dire nous montrons comment à un instant donné un agent l'utilise pour choisir le(s) partenaire(s) qu'il croit le(s) plus susceptible(s) d'accepter sa proposition de formation de coalition. Par ailleurs, nous croyons que notre modèle comporte aussi un aspect dynamique très intéressant. Dans cette section, nous montrons qu'un mécanisme de raisonnement social peut permettre à un agent de faire *évoluer ses choix*. Dans une société ouverte, les agents peuvent entrer et sortir du système à n'importe quel moment. Ainsi les agents peuvent prendre des décisions différentes à des moments différents concernant les partenaires à qui envoyer des propositions de formation de coalition.

En ce qui concerne les buts et les plans, l'entrée d'un agent qui est capable d'entreprendre une certaine action peut rendre cette action disponible. Par conséquent, elle peut aussi rendre à la fois un plan exécutable et un but réalisable, comme expliqué dans la section 6.1. Nous avons illustré implicitement cette situation dans la section 7.1, lorsque l'arrivée de deux agents dans l'exemple du laboratoire de recherche a rendu le but $write\_ss\_mas\_paper$ réalisable pour l'agent $ag2$.

Afin d'approfondir l'analyse de l'évolution du choix de partenaires, nous utilisons un exemple du monde des blocs, proposé initialement dans [SD95b], et dont la situation initiale est présentée dans la figure 7.5.

Dans cet exemple, la société est composée de trois agents : $ag1$, $ag2$ et $ag3$. Ces agents ont pour but d'empiler des blocs les uns sur les autres et de les dépiler. Les actions disponibles sont empiler (put_on) et dépiler ($clear$). Les blocs A, B et C sont considérés comme des ressources.

Nous allons analyser l'évolution du choix de partenaires en supposant de faibles changements, considérés comme ayant lieu à des instants différents. Pour chacun de ces instants, nous présentons les relations et les situations de dépendances

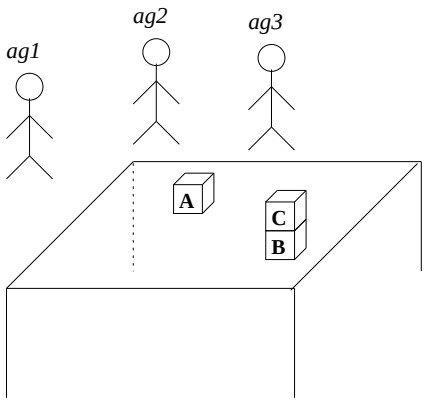


Figure 7.5 : Premier exemple du monde des blocs

entre les agents. Dans les tableaux suivants, les cases grisées représentent les changements par rapport à l'instant précédent. Nous considérons aussi que (i) tous les agents sont capables de percevoir ces changements et de mettre à jour leurs descriptions externes respectives et (ii) tous les agents ont, deux à deux, les mêmes représentations l'un de l'autre. Nous ne considérons que les choix de partenaires fait par l'agent $ag3$, qui tient lieu du proposant.

7.3.1 Scénario à t=0

La description externe correspondant à la situation initiale de cette société est présentée dans l'exemple 7.1. L'agent $ag3$ infère par exemple la situation de but et la relation de dépendance suivantes :

$$DEP(ag3, on(A,B))$$
$$dep\_on_a(ag3, ag1, on(A,B), ag3)$$

Exemple 7.1 : Description externe à t=0.

Agent	Buts	Actions	Ressources	Plans
$ag1$	on(C,table)	put_on	B	on(C,table):=clear(C).
$ag2$	on(A,B)	—	C	on(A,B):=clear(C), put_on(A,B).
$ag3$	on(A,B)	clear	A	on(A,B):=clear(C), put_on(A,B).

Les relations de dépendance sont présentées dans l'exemple 7.2 et les situations de dépendance dans l'exemple 7.3. Chaque ligne dans ces tables représente le point de vue de l'agent situé dans la première colonne, noté par "moi". Par exemple, la relation de dépendance précédente est représentée dans la troisième ligne de l'exemple 7.2.

Exemple 7.2 : Relations de dépendance à t=0.

R-DEP		Agents		
moi	but	$ag1$	$ag2$	$ag3$
$ag1$	$on(C, table)$	—	—	$clear$
$ag2$	$on(A, B)$	put_on	—	$clear$
$ag3$	$on(A, B)$	put_on	—	—

Exemple 7.3 : Situations de dépendance à t=0.

D-SIT		Agents		
moi	but	$ag1$	$ag2$	$ag3$
$ag1$	$on(C, table)$	—	IND	UD
$ag2$	$on(A, B)$	UD	—	UD
$ag3$	$on(A, B)$	UD	IND	—

Dans ce scénario, chaque agent n'a des plans que pour son propre but. Par conséquent, tous les agents déduisent une *UD*. Notons qu'un observateur extérieur n'arriverait pas à la même conclusion. Si à cet instant l'agent $ag3$ devait proposer à l'agent $ag1$ de former une coalition, le contenu du message envoyé serait le suivant :

"J'ai détecté une UD entre nous, car j'ai besoin de ton action put_on pour atteindre mon but on(A,B). Est-ce que tu veux l'entreprendre ?"

Comme l'agent $ag3$ a déduit une *UD*, il ne peut offrir aucune action en échange qui puisse intéresser l'agent $ag1$. Ses chances d'obtenir l'action désirée dépendent fortement de ce que l'agent $ag1$ est en train de faire à cet instant, du comportement bienveillant de ce dernier, etc. En outre, nous pouvons noter que dans ce scénario l'agent $ag3$ *n'a aucun autre choix possible de partenaire* pour atteindre son but.

7.3.2 Scénario à t=1

Supposons qu'à t=1, l'agent $ag3$ apprenne un plan pour dépiler un bloc, tout en n'ayant aucun but de ce genre. La manière dont cet apprentissage peut être réalisé est hors de propos ici. Nous considérons seulement qu'il a ajouté cette information dans sa description externe. La nouvelle description externe et les nouvelles situations de dépendance qui en découlent sont présentées dans les exemples 7.4 et 7.5. Etant donné qu'aucun nouvel agent n'est entré dans la société et qu'aucun de ceux qui en font partie n'ont changé leurs actions et ressources, les relations de dépendance sont les mêmes que celles présentées dans l'exemple 7.2.

Exemple 7.4 : Description externe à t=1.

Agent	Buts	Actions	Ressources	Plans
$ag1$	on(C,table)	put_on	B	on(C,table):=clear(C).
$ag2$	on(A,B)	—	C	on(A,B):=clear(C), put_on(A,B).
$ag3$	on(A,B)	clear	A	on(A,B):=clear(C), put_on(A,B). on(C,table):=clear(C).

Exemple 7.5 : Situations de dépendance à t=1.

D-SIT		Agents		
moi	but	$ag1$	$ag2$	$ag3$
$ag1$	$on(C, table)$	—	*IND*	*UD*
$ag2$	$on(A, B)$	*UD*	—	*UD*
$ag3$	$on(A, B)$	*LBRD*	*IND*	—

Nous pouvons observer que dans ce scénario, l'agent $ag3$ infère une *LBRD* envers l'agent $ag1$. Il peut donc envoyer maintenant une proposition d'échange social pour obtenir l'action qu'il désire. Si à cet instant l'agent $ag3$ devait commencer une interaction avec l'agent $ag1$, son contenu serait le suivant :

"J'ai détecté une LBRD entre nous. En ce qui me concerne, j'ai besoin de ton action put_on pour atteindre mon but on(A,B). Par contre, je crois que toi aussi tu as aussi besoin de mon action clear pour atteindre ton but on(C,table). Si tu m'aides, je t'aiderai après."

La stratégie de l'agent $ag3$ est d'abord de convaincre l'agent $ag1$ que celui-ci a besoin de lui pour atteindre le but $on(C, table)$. Ceci n'est pas très difficile, puisque, en ayant inféré une *UD*, ce dernier en est conscient. Ce qu'en revanche l'agent $ag1$ ne connaît pas à cet instant, c'est que l'agent $ag3$ dépend aussi de lui pour atteindre le but $on(A, B)$.

7.3.3 Scénario à t=2

Supposons qu'à t=2, l'agent $ag1$ apprenne lui aussi un plan pour empiler des blocs. La nouvelle description externe et les nouvelles situations de dépendance sont présentées respectivement dans les exemples 7.6 et 7.7. Les relations de dépendance sont les mêmes que celles présentées dans l'exemple 7.2.

Exemple 7.6 : Description externe à t=2.

Agent	Buts	Actions	Ressources	Plans
$ag1$	on(C,table)	put_on	B	on(C,table):=clear(C). on(A,B):=clear(C), put_on(A,B).
$ag2$	on(A,B)	—	C	on(A,B):=clear(C), put_on(A,B).
$ag3$	on(A,B)	clear	A	on(A,B):=clear(C), put_on(A,B). on(C,table):=clear(C).

Exemple 7.7 : Situations de dépendance à t=2.

D-SIT		Agents		
moi	but	$ag1$	$ag2$	$ag3$
$ag1$	$on(C, table)$	—	*IND*	*MBRD*
$ag2$	$on(A, B)$	*UD*	—	*UD*
$ag3$	$on(A, B)$	*MBRD*	*IND*	—

Nous pouvons noter que dans ce scénario, les deux agents $ag3$ et $ag1$ infèrent une *MBRD* l'un envers l'autre. Par conséquent les deux agents sont conscients qu'un échange social peut avoir lieu. Cette situation amène à une interaction plus rapide et facile, car aucun d'entre eux n'a besoin de réfléchir à l'avantage de s'engager dans une activité commune. En outre, comme chacun d'entre eux est conscient que l'autre dépend de lui, les chances de réciprocité augmentent. Dans le scénario précédent, comme l'agent $ag1$ n'avait pas de plan pour le but $on(A, B)$, il pouvait penser que l'agent $ag3$ n'était pas vraiment intéressé à l'aider : par exemple, il pouvait penser que celui-ci a utilisé l'argument d'échange social seulement pour le convaincre d'entreprendre l'action qui l'intéressait. Ce dernier aspect consiste à supposer que non seulement les agents ne sont pas nécessairement bienveillants, mais qu'ils peuvent aussi ne pas être *sincères*, c'est-à-dire, qu'en profitant de leur pouvoir social, ils peuvent délibérément choisir de donner une information incorrecte aux autres avec le but de les exploiter. Dans cette thèse, nous ne nous occupons pas de tels types de interactions sociales, les ayant exclus par le principe de la sincérité (P2). Dans nos perspectives, décrites dans la section 12.2.3, nous discutons cet aspect en plus de détail.

L'interaction est similaire à celle présentée dans la section précédente. La seule différence réside dans le fait que maintenant le destinataire est conscient de la dépendance du proposant envers lui-même. Il sera ainsi plus susceptible d'accepter de participer dans la coalition.

7.3.4 Scénario à t=3

Supposons qu'à t=3, l'agent $ag4$ arrive, avec le but $put\_on(A, B)$ et un plan pour l'atteindre. Ce nouveau scénario est présenté dans les exemples 7.8, 7.9 et 7.10.

Exemple 7.8 : Description externe à t=3.

Agent	Buts	Actions	Ressources	Plans
$ag1$	on(C,table)	put_on	B	on(C,table):=clear(C). on(A,B):=clear(C), put_on(A,B).
$ag2$	on(A,B)	—	C	on(A,B):=clear(C), put_on(A,B).
$ag3$	on(A,B)	clear	A	on(A,B):=clear(C), put_on(A,B). on(C,table):=clear(C).
$ag4$	on(A,B)	put_on	—	on(A,B):=put_on(A,B).

Exemple 7.9 : Relations de dépendance à t=3.

R-DEP		Agents			
moi	but	$ag1$	$ag2$	$ag3$	$ag4$
$ag1$	$on(C, table)$	—	—	$clear$	—
$ag2$	$on(A, B)$	put_on	—	$clear$	put_on
$ag3$	$on(A, B)$	put_on	—	—	put_on
$ag4$	$on(A, B)$	—	—	—	—

Exemple 7.10 : Situations de dépendance à t=3.

D-SIT		Agents			
moi	but	$ag1$	$ag2$	$ag3$	$ag4$
$ag1$	$on(C, table)$	—	IND	$MBRD$	IND
$ag2$	$on(A, B)$	UD	—	UD	UD
$ag3$	$on(A, B)$	$MBRD$	IND	—	$LBMD$
$ag4$	$on(A, B)$	IND	IND	IND	—

Comme la société a maintenant un nouveau membre qui peut entreprendre une action qui intéresse les autres, les relations de dépendance changent. S'ajoutant aux résultats présentés dans la section 7.3.1, l'agent $ag3$ peut inférer une nouvelle relation de dépendance :

$$dep\_on_a(ag3, ag4, on(A, B), ag3)$$

Notons que maintenant l'agent $ag3$ *a un choix* pour obtenir l'action put_on, dans la mesure où, soit l'agent $ag1$ soit l'agent $ag4$ peuvent l'entreprendre. Par ailleurs, pour l'instant, l'agent $ag4$ croit qu'il peut atteindre son but tout seul, c'est-à-dire qu'il déduit la situation de but suivante :

$$AUT(ag4, on(A, B))$$

L'agent $ag3$ infère maintenant une $LBMD$ envers l'agent $ag4$, car en utilisant ses plans il infère que les deux dépendent l'un de l'autre pour atteindre le but $on(A, B)$. Par conséquent, il est conscient qu'une coopération peut avoir lieu. S'il décide de proposer à l'agent $ag4$ de former une coalition, le contenu du message envoyé serait le suivant :

"J'ai calculé une LBMD entre nous pour le but on(A,B), car je suppose que tu as besoin de moi pour entreprendre l'action clear. Comme je ne peux pas entreprendre l'action put_on, je te propose de coopérer avec moi pour atteindre notre but commun."

De plus, l'agent $ag3$ a toujours l'option de proposer un échange à l'agent $ag1$. Comme ces deux situations de dépendance ($LBMD$ et $MBRD$) ne sont pas comparables selon le critère $\prec_{partner}$, un choix aléatoire doit être fait pour choisir à quel partenaire s'adresser.

7.3.5 Scénario à t=4

Dans ce dernier scénario, l'agent $ag4$ apprend que son plan était incomplet et qu'il a besoin de l'action $clear(C)$ pour atteindre son but $on(A, B)$. Ce nouveau scénario est présenté dans les exemples 7.11, 7.12 et 7.13. Les relations et situations de dépendance vont changer, car maintenant, l'agent $ag4$ infère la situation de but et la relation de dépendance suivantes :

$$DEP(ag4, on(A, B))$$
$$dep\_on_a(ag4, ag3, on(A, B), ag4)$$

Exemple 7.11 : Description externe à t=4.

Agent	Buts	Actions	Ressources	Plans
$ag1$	on(C,table)	put_on	B	on(C,table):=clear(C). on(A,B):=clear(C), put_on(A,B).
$ag2$	on(A,B)	—	C	on(A,B):=clear(C), put_on(A,B).
$ag3$	on(A,B)	clear	A	on(A,B):=clear(C), put_on(A,B). on(C,table):=clear(C).
$ag4$	on(A,B)	put_on	—	on(A,B):=clear(C), put_on(A,B).

Exemple 7.12 : Relations de dépendance à t=4.

R-DEP		Agents			
moi	but	$ag1$	$ag2$	$ag3$	$ag4$
$ag1$	$on(C, table)$	—	—	$clear$	—
$ag2$	$on(A, B)$	put_on	—	$clear$	put_on
$ag3$	$on(A)$	put_on	—	—	put_on
$ag4$	$on(A)$	—	—	$clear$	—

Exemple 7.13 : Situations de dépendance à t=4.

D-SIT		Agents			
moi	but	$ag1$	$ag2$	$ag3$	$ag4$
$ag1$	$on(C, table)$	—	IND	$MBRD$	IND
$ag2$	$on(A, B)$	UD	—	UD	UD
$ag3$	$on(A, B)$	$MBRD$	IND	—	$MBMD$
$ag4$	$on(A, B)$	IND	IND	$MBMD$	—

Nous pouvons noter que maintenant les deux agents $ag3$ et $ag4$ infèrent une $MBMD$ l'un envers l'autre. Ces deux agents peuvent donc offrir en échange une action qui intéresse l'autre. Cette situation est similaire à celle présentée dans la section 7.3.3. La différence majeure est que le but que doivent accomplir les deux

agents est maintenant le même. La discussion sur la réciprocité est également valable ici, mais maintenant celle-ci est plus probable, car les deux agents ont le même but. Finalement, en ce qui concerne le choix de partenaires de l'agent $ag3$, comme une *MBMD* est la meilleure situation de dépendance possible par rapport à notre critère $\prec_{partner}$, il va choisir d'interagir avec l'agent $ag4$.

7.4 Discussion

Dans ce chapitre, nous avons montré comment un mécanisme de raisonnement social peut être exploité par un agent pour choisir un ou bien accepter des partenaires pour une formation de coalition. Nous avons proposé d'utiliser les situations de dépendance comme un critère de décision qui permet au proposant d'évaluer la possibilité que le destinataire a d'accepter une telle proposition.

En ce qui concerne l'évolution de ces choix, nous avons montré un aspect important du processus de formation de coalitions : *en prenant compte des informations que les agents ont les uns des autres, ils peuvent s'adapter aux sociétés ouvertes, en choisissant des partenaires différents s'ils croient que les chances d'obtenir un comportement coopératif de ces derniers sont plus grandes.*

Plusieurs travaux ont déjà abordé le choix de partenaires[7]. Notre approche pour ces critères de décision est différente de celle adoptée dans le cadre des théories de jeux, comme dans [GD93, RZ94]. Dans celles-ci, le choix est fondé seulement sur une fonction de maximisation de l'utilité. Nous avons montré que de telles approches ne sont pas suffisantes pour caractériser des sociétés ouvertes : elles adoptent implicitement que tous les buts sont réalisables et que tous les plans sont exécutables, sans prendre en compte la dynamique de la société d'agents.

Une autre approche pour le choix de partenaires qui utilise aussi la notion de dépendance est présentée dans [CM92, CM93]. Dans ces travaux, les auteurs définissent un critère appelé "willingness" pour exprimer la possibilité d'un agent de collaborer avec d'autres. Ce critère correspond en quelque sorte à notre notion de dépendance mutuelle. Par contre, les dépendances réciproques (et par conséquent l'interaction d'échange social) n'y sont pas abordées. A la différence de notre approche, les buts sociaux de coopérer et d'échanger sont représentés en termes de buts au sein d'un agent, et non comme des stratégies, comme dans notre cas. Par ailleurs, nous n'avons pas connaissance d'une implémentation de tel modèle.

Dans [Mar93], un modèle de confiance entre agents est présenté. Selon l'auteur, la confiance est une notion associée au risque : on est confiant lorsqu'on a beaucoup à perdre et peu à gagner. Ce modèle présent des divers types de confiance (générale ou pour une tâche particulière), chacun avec différents niveaux quantitatifs. A partir de ses valeurs, les agents calculent les possibles bénéfices et risques

[7] Même si l'expression pour désigner cette activité soit parfois différente. Par exemple, dans la théorie des jeux, on parle souvent de l'émergence de la coopération.

de l'activité commune et décident de coopérer ou non. Cette notion de confiance peut être rapprochée à nos critères de choix de partenaires.

Chapitre 8

Effet sur la révision de croyances

Dans ce chapitre, nous analysons l'effet du mécanisme de raisonnement social sur la révision de croyances. En particulier, nous montrons qu'en utilisant un tel mécanisme, un agent peut détecter que sa représentation des autres agents est incomplète ou incorrecte. Ce chapitre reprend fondamentalement les idées que nous avons initialement présentées dans [SD95a].

Tout d'abord, nous utilisons le principe de l'auto-connaissance (P3) introduit dans la section 1.3 : nous considérons que les agents *connaissent* leurs propres propriétés, mais ils ont des *croyances* sur les propriétés des autres. L'utilisation des croyances pour modéliser les autres paraît un choix raisonnable, à partir du moment où nous considérons que la source de telles informations peuvent être erronées, comme nous l'avons discuté dans la section 5.1.2.

Ce chapitre est organisé de la façon suivante. Nous commençons par présenter les approches existantes pour modéliser les croyances dans la section 8.1, et nous justifions notre choix pour le modèle déductif de croyances [Kon86]. Nous définissons notre langage externe dans la section 8.2, en particulier l'hypothèse de compatibilité de description externe. Nous utilisons cette hypothèse pour démontrer dans la section 8.3 que dans quelques situations, le fait que deux agents calculent deux situations de dépendance différentes l'un envers l'autre pour un même but signifie que leurs descriptions externes ne sont pas compatibles. Nous approfondissons cette analyse en démontrant que ce résultat est expliqué par des croyances incomplètes ou fausses qu'un agent peut avoir de l'autre. Une analyse exhaustive de tous les résultats couplés possibles de deux mécanismes de raisonnement social est également présentée. Ensuite, dans la section 8.4, nous montrons comment cette problématique peut être liée à celle de la révision de croyances et nous proposons une solution très simple pour le choix de contexte dans la section 8.5. Enfin, dans la section 8.6, nous terminons ce chapitre par une discussion sur les résultats obtenus.

8.1 Modélisation des croyances

Afin de pouvoir comparer des résultats obtenus par les mécanismes de raisonnement social de deux agents, nous devons nous placer comme un observateur extérieur à la société pour analyser les résultats inférés par chacun. Il faut donc introduire un langage externe, qui puisse identifier les agents sujet, le sens de ce terme étant défini dans le chapitre 5.

Techniquement parlant, nous devons pour cela pouvoir représenter, d'un point de vue externe, les *croyances* d'un agent. Dans la sous-section 8.1.1, nous présentons les approches existantes pour modéliser les croyances. Nous justifions notre choix pour le modèle déductif de croyances [Kon86] dans la sous-section 8.1.2. Une description détaillée de ce modèle est présentée ensuite dans la sous-section 8.1.3.

8.1.1 Approches existantes

Selon [WJ94], un formalisme pour exprimer des croyances doit traiter deux attributs indépendants : la syntaxe du langage et son modèle sémantique.

En ce qui concerne l'aspect syntaxique, deux approches sont possibles :

- l'approche *modale*, où un ensemble d'opérateurs modaux est défini. Ces opérateurs sont appliqués aux formules auxquelles nous voulons attribuer la notion de croyance ;
- l'approche *méta-linguistique*, où nous utilisons un langage de premier ordre avec sorte qui contient des termes qui désignent des formules d'un autre langage objet.

Pour l'aspect sémantique, deux approches sont également possibles :

- l'approche des *mondes possibles*, proposée initialement par Hintikka [Hin62], où les croyances sont caractérisées par les mondes possibles et où une relation d'accessibilité est définie entre ces mondes. Nous pouvons y associer une théorie de correspondance, ce qui rend ce modèle attractif du point de vue mathématique. Par contre, un des problèmes les plus connus de ce modèle est celui de l'*omniscience logique*, c'est-à-dire les agents sont considérés comme des raisonneurs parfaits ;
- l'approche *sententielle* ou de *structures symboliques interprétées*, où les croyances sont considérées comme des formules présentes dans la base de croyances d'un agent.

8.1.2 Approche suivie

Nous avons choisi d'utiliser le modèle déductif de croyances défini dans [Kon86]. En ce qui concerne l'aspect syntaxique, le modèle utilise des opérateurs modaux, et suit donc la première approche. Du point de vue sémantique, ce modèle suit

la deuxième approche : les agents peuvent dériver des conclusions à partir d'un ensemble initial de croyances, sans imposer qu'ils soient des raisonneurs parfaits, c'est-à-dire, qu'ils ne dérivent pas nécessairement toutes les conclusions logiques de cet ensemble initial.

Nous avons choisi ce modèle car, à notre avis, il fournit un cadre plus opérationnel pour représenter les agents qui nous intéressent. En particulier, il n'est pas nécessaire de considérer les agents comme étant des raisonneurs parfaits.

8.1.3 Modèle déductif de croyances

Dans la figure 8.1, nous présentons un système de croyances M. Ses composantes sont un ensemble de croyances de base, des règles d'inférence et une stratégie de contrôle. Lorsqu'on lui pose une question, le système répond dans un temps borné, soit directement en mettant la question en correspondance avec un fait appartenant à son ensemble de croyances de base, soit indirectement en appliquant des règles d'inférence à ce même ensemble. L'ensemble de croyances bel(M) est défini comme étant l'ensemble de formules fermées pour lesquelles le système de croyances M répond "oui". Techniquement, l'expression "Ralph croit en ϕ" signifie que $\phi \in$ bel(Ralph).

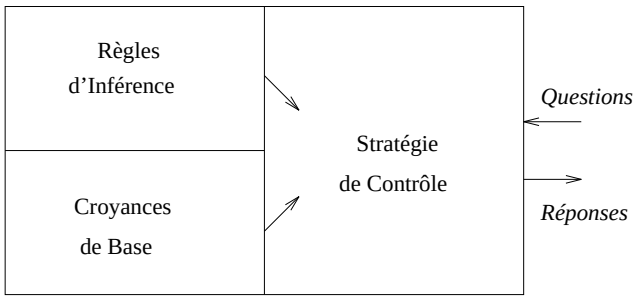


Figure 8.1 : Modèle déductif de croyances [Kon86]

Dans ce modèle, l'ensemble de croyances bel(M) est considéré comme clos pour la déduction, c'est-à-dire, la stratégie de contrôle applique de façon exhaustive les règles de déduction. Ces dernières correspondent à des règles d'inférence correctes, qui peuvent être effectivement calculées et qui ont un nombre fixe et fini de prémisses. Par exemple, les règles d'inférence classiques du calcul de prédicats de premier ordre, comme le Modus Ponens, sont des règles de déduction.

Pour définir formellement un système de croyances, on définit une structure mathématique appelée *structure de déduction*, qui est composée de deux ensembles $\langle B, R \rangle$. L'ensemble B est un ensemble de formules fermées dans un langage logique L (qui correspond à l'ensemble de croyances de base) et l'ensemble R est un ensemble de règles de déduction. L'ensemble de croyances d'une structure de déduction $\langle B, R \rangle$ est composé de ses croyances de base B et de toutes les formules fermées dérivables à partir de ces dernières en utilisant des règles de déduction R :

$$\mathsf{bel}(\langle\, \mathsf{B}, \mathsf{R} \,\rangle) \stackrel{\text{def}}{=} \{\phi \mid B \vdash_R \phi\}$$

Le langage L est le *langage interne* de croyances, qui est différent du *langage externe*. Ce dernier est utilisé par un observateur extérieur pour décrire les croyances du système. A partir d'un langage interne de premier ordre L_0, avec des constantes mais sans symboles fonctionnels, et d'un ensemble énumérable d'agents $\{S_0, S_1, \ldots\}$, on construit le langage externe L^B de la façon suivante :

1. Toutes les formules fermées et règles de formation de L_0 appartiennent aussi à L^B ;
2. Si ϕ est une formule fermée de L_0, alors $[S_i]\,\phi$ est une formule de L^B.

Formellement, un modèle de L^B, dénoté par $B(L_0, \rho)$, ayant comme paramètres le langage interne L_0 et une fonction ρ qui donne pour chaque agent i l'ensemble de ses règles de déduction $\rho(i)$, est défini comme :

$$B(L_0, \rho) \stackrel{\text{def}}{=} \langle\, \varphi, v_0, \mathsf{U}, D, \eta \,\rangle$$

où

- $\langle\, \varphi, V_0, \mathsf{U} \,\rangle$ est une interprétation standard pour le langage interne L_0 ;
- $D = \{d_0, d_1, \ldots\}$ est une séquence de structures de déduction, une pour chaque agent, dont toutes utilisent le même langage interne L_0 et dont les règles de déduction de d_i sont données par $\rho(i)$;
- η est un ensemble de fonctions de nommage[1] η_i qui retourne pour chaque agent S_i un symbole de constante pour chaque élément de l'univers de discours U.

Si m est un modèle de L^B, la validité d'une formule avec l'opérateur de croyance dans m est définie par :

[1] Pour montrer l'intérêt des fonctions de nommage, prenons un exemple proposé dans [Kon86]. Appelons HP le fameux inspecteur Hercule Poirot et soient les deux formules suivantes : $[HP]\exists x\ murderer(x)$ et $\exists x\ [HP] murderer(x)$. L'interprétation de la première est que l'inspecteur croit qu'il existe quelqu'un qui est l'assassin, tandis que l'interprétation de la deuxième est qu'il croit savoir qui est cet assassin. En ce qui concerne la base de croyances $\mathsf{bel}(\mathsf{HP})$ de l'inspecteur, dans le premier cas nous avons effectivement $\exists x\ murderer(x) \in \mathsf{bel}(\mathsf{HP})$. Ce n'est pas le cas en ce qui concerne la deuxième formule, car x est une variable libre dans $murderer(x)$. Dans ce dernier, nous avons $murderer(c) \in \mathsf{bel}(\mathsf{HP})$, où c est un symbole de constante qui fait référence à l'individu x. Dans le langage externe, les deux formules $\exists x\ [HP] murderer(x)$ et $[HP] murderer(c)$ disent la même chose sur les croyances de l'inspecteur, tandis que c fait référence au même individu que x. Konolige appelle *quantification interne* ce concept syntaxique consistant à utiliser une variable liée en dehors de la portée de l'opérateur de croyances et d'y faire référence dans le contexte de ce même opérateur. C'est justement pour représenter cette situation qu'il propose dans son modèle l'utilisation des fonctions de nommage, dénoté par η. En effet, il propose même un deuxième langage externe L^{Bq}, en utilisant un nouvel opérateur $\bullet$ pour représenter ces constantes. Nous n'avons pas cependant utilisé ce langage, car dans notre contexte nous supposons que les agents utilisent des noms standard.

$$\models_m [S_i]\,\phi \text{ ssi } \phi \in \mathsf{bel}(\mathsf{d_i}).$$

Dans notre modèle, nous supposons l'existence de *noms standard* pour tous les éléments de l'univers de discours que nous traitons. Cela veut dire que tous les agents utilisent le même symbole de constante pour désigner dans leur langage interne le même élément de l'univers de discours. Par exemple, étant donné un but g de l'univers et un ensemble d'agents $\{i, j, \ldots\}$ nous aurons $\eta_i(\mathsf{g}) = \eta_j(\mathsf{g}) = \ldots = g$.

8.2 Langage externe

Nous avons adopté le modèle déductif de croyances décrit ci-dessus pour formaliser notre langage externe. Nous avons néanmoins utilisé une notation syntaxique différente pour les opérateurs de croyances : pour un agent i, nous notons $\mathbf{B_i}$ au lieu de $[i]$ l'opérateur de croyances respectif. Autrement dit, nous avons défini un ensemble d'opérateurs de croyances $\{\mathbf{B_i}, \mathbf{B_j}, \ldots\}$ de telle sorte que la formule dans notre langage externe $\mathbf{B_i}\,\phi$ signifie que la formule ϕ appartient à la base de croyances $\mathsf{bel(i)}$ de l'agent i.

Nous ne sommes a priori intéressés que par les résultats des mécanismes de raisonnement social de deux agents. Les formules exprimées dans le langage interne que nous allons traiter se limiteront donc à celles que ceux-ci peuvent inférer, décrites dans le chapitre 5. Dans les sections suivantes, nous montrons les aspects essentiels de ce langage externe.

8.2.1 Notions de base

Tout d'abord, nous avons besoin d'exprimer le fait que deux agents i et j quelconques ont la même entrée de description externe relative à un troisième agent k. Pour cela, nous introduisons les quatre prédicats suivants qui expriment respectivement que les agents i et j ont les mêmes croyances en ce qui concerne les buts, les actions, les ressources et les plans de l'agent k :

$$Comp_g(i,j,k) \Leftrightarrow \forall g\ (\mathbf{B_i}\ is_g(k,g) \Leftrightarrow \mathbf{B_j}\ is_g(k,g)) \tag{8.1}$$

$$Comp_a(i,j,k) \Leftrightarrow \forall a\ (\mathbf{B_i}\ is_a(k,a) \Leftrightarrow \mathbf{B_j}\ is_a(k,a)) \tag{8.2}$$

$$Comp_r(i,j,k) \Leftrightarrow \forall r\ (\mathbf{B_i}\ is_r(k,r) \Leftrightarrow \mathbf{B_j}\ is_r(k,r)) \tag{8.3}$$

$$Comp_p(i,j,k) \Leftrightarrow \forall p\ (\mathbf{B_i}\ is_p(k,p) \Leftrightarrow \mathbf{B_j}\ is_p(k,p)) \tag{8.4}$$

A partir des prédicats précédents, le fait que les agents i et j ont la même entrée de description externe relative à l'agent k est exprimé par :

$$Comp(i,j,k) \Leftrightarrow Comp_g(i,j,k) \wedge Comp_a(i,j,k) \wedge Comp_r(i,j,k) \wedge Comp_p(i,j,k) \tag{8.5}$$

Comme nous venons d'expliquer dans la section précédente, il nous est possible d'utiliser la quantification interne parce que nous supposons l'existence de noms standard.

8.2.2 Hypothèse de compatibilité de description externe

Nous appelons *hypothèse de compatibilité de description externe* le fait que deux agents aient les mêmes entrées de descriptions externes relatives à chacun d'entre eux. Autrement dit, cette hypothèse signifie que deux agents ont des croyances complètes et correctes l'un de l'autre :

$$Ext_c(i,j) \quad \Leftrightarrow \quad Comp(i,j,i) \wedge Comp(i,j,j) \tag{8.6}$$

8.3 Inconsistance au niveau de la société

Dans les chapitres 6 et 7, nous nous sommes intéressés à l'analyse des effets du mécanisme de raisonnement social sur ses choix de buts, plans et partenaires. Pour cela, dans les exemples présentés, nous avons implicitement adopté *l'hypothèse de compatibilité de description externe*. Autrement dit, nous avons considéré que les informations que chaque agent possède sur les autres sont à la fois *complètes* et *correctes*. Comme nous l'avons exposé dans la section 5.1.2, cette situation ne correspond évidemment pas au cas général, si nous supposons que les moyens par lesquels un agent peut acquérir ces informations, comme la perception et la communication, peuvent générer des erreurs.

Nous appelons *inconsistance au niveau de la société* le fait que deux agents aient des entrées de description externe différentes relatives à chacun d'entre eux. Nous allons montrer dans la suite que cette inconsistance peut être détectée en analysant les résultats des mécanismes de raisonnement social de deux agents.

8.3.1 Analyse préliminaire des résultats couplés

Nous appelons *résultat couplé* une paire de résultats obtenus par les mécanismes de raisonnement social de deux agents différents, en ce qui concerne leurs situations de but et de dépendance. Dans cette section, nous montrons que quelques résultats couplés particuliers sont des indicateurs d'inconsistance au niveau de la société.

Pour cela, nous prenons tout d'abord comme hypothèse que tous les agents calculent leurs situations de but et de dépendance tel que nous les avons définies dans le chapitre 5. Ainsi, l'utilisation de l'opérateur de croyances $\mathbf{B_i}$ dans une formule comme $\mathbf{B_i}\ MBMD(i,j,g)$ exprime que l'agent sujet est l'agent i, et que la description externe utilisée pour calculer cette situation de dépendance est celle appartenant à l'agent i.

En utilisant la définition de compatibilité de description externe décrite dans l'équation 8.6, nous démontrons facilement les théorèmes suivants :

$$\mathbf{B_i}\ MBMD(i,j,g) \wedge \neg \mathbf{B_j}\ MBMD(j,i,g) \quad \Rightarrow \quad \neg Ext_c(i,j) \tag{8.7}$$

$$\mathbf{B_i}\ MBRD(i,j,g,g') \wedge \neg \mathbf{B_j}\ MBRD(j,i,g',g) \quad \Rightarrow \quad \neg Ext_c(i,j) \tag{8.8}$$

$$\mathbf{B_i}\ LBMD(i,j,g) \wedge \mathbf{B_j}\ LBMD(j,i,g) \quad \Rightarrow \quad \neg Ext_c(i,j) \tag{8.9}$$

$$\mathbf{B_i}\ LBRD(i,j,g,g') \wedge \mathbf{B_j}\ LBRD(j,i,g',g) \quad \Rightarrow \quad \neg Ext_c(i,j) \tag{8.10}$$

Les preuves de ces théorèmes sont présentées dans l'annexe A[2]. Prenons par exemple le théorème 8.7. Si l'agent i déduit une *MBMD* envers l'agent j pour un certain but g, cela signifie qu'il a les croyances suivantes :

1. l'un comme l'autre a ce but à atteindre et les deux ont au moins un plan dont l'exécution atteint ce but ;
2. en prenant son plan, il est capable d'entreprendre une action nécessaire à son exécution qui ne peut pas être réalisée par l'agent j ;
3. il existe également une action nécessaire dans ce plan qu'il est incapable d'entreprendre mais que l'agent j peut faire ;
4. les deux conclusions précédentes peuvent être également inférées en utilisant le plan de j.

Si nous prenons l'hypothèse de compatibilité de description externe comme valable, ces propriétés sont préservées dans la description externe de j, qui devrait donc inférer lui aussi une *MBMD*.

Nous représentons ces résultats dans la table 8.1. Un "xxx" signifie qu'une inconsistance existe au niveau de la société. Cette table est incomplète, dans le sens où quelques cases qui ne sont pas marquées comme inconsistantes au niveau de la société peuvent l'être, en adoptant des hypothèses supplémentaires, telles, par exemple que les agents ont des mécanismes de planification similaires. C'est ce que nous montrons dans la section 8.3.3. Par contre, les cases indiquées présentent une condition suffisante pour une inconsistance au niveau de la société.

D-SIT	*agent j*					
agent i	*IND*	*UD*	*LBRD*	*LBMD*	*MBRD*	*MBMD*
IND					xxx	xxx
UD					xxx	xxx
LBRD			xxx		xxx	xxx
LBMD				xxx	xxx	xxx
MBRD	xxx	xxx	xxx	xxx		xxx
MBMD	xxx	xxx	xxx	xxx	xxx	

Table 8.1 : Inconsistance au niveau de la société

8.3.2 Croyances incomplètes et croyances fausses

Nous avons montré jusqu'à présent que dans des circonstances particulières, si deux agents déduisent des situations de dépendance différentes l'un envers l'autre

[2]Dans l'annexe A, ce résultats correspondent respectivement aux corollaires A.10.1, A.10.2, A.10.3 et A.10.4.

pour un même but, leurs descriptions externes sont incompatibles, c'est-à-dire, il existe une inconsistance au niveau de la société. Dans cette section, nous approfondissons cet aspect, en essayant de comprendre quel genre d'incompatibilité existe.

Pour être capable de réaliser cette analyse, nous avons adopté une hypothèse restrictive supplémentaire : nous supposons que les agents ont le *même mécanisme de planification*, c'est-à-dire que les plans générés par chacun d'entre eux sont strictement les mêmes. Cela signifie que la proposition suivante est vraie :

$$Comp_p(i,j,i) \wedge Comp_p(i,j,j)$$

Par conséquent, aucun agent ne va inférer des dépendances localement reconnues (soit mutuelles soit réciproques). Nous nous concentrons aussi sur l'analyse des inconsistances relatives aux actions que chaque agent croit que l'autre peut entreprendre[3].

Nous rappelons que la notion centrale de notre mécanisme de raisonnement social est celle de relation de dépendance. Cette notion signifie que les agents peuvent *avoir besoin* ou *offrir* des actions (utilisées dans un plan) les uns aux autres. En effet, nous avons prouvé les théorèmes suivants[4] :

$$dep\_on_a(i,j,g,k) \Rightarrow \exists a \ (\neg is_a(i,a) \wedge is_a(j,a)) \tag{8.11}$$

$$\forall a \ (is_a(i,a) \vee \neg is_a(j,a)) \Rightarrow \neg dep\_on_a(i,j,g,k) \tag{8.12}$$

Actions requises

Quand un agent i infère le prédicat $dep\_on_a(i,j,g,i)$, son interprétation de cette formule est la suivante, selon l'équation 8.11 :

En utilisant mes plans, je crois que je dépends de toi pour atteindre le but g car il existe une action nécessaire dans un plan atteignant ce but, que tu peux entreprendre et que je ne peux pas entreprendre.

Appelons cette action une *action requise*. Supposons que l'agent j ne puisse pas entreprendre cette action. Dans ce cas, nous aurons une inconsistance au niveau de la société, dont la cause est l'existence au sein d'un agent, de *croyances fausses relatives à ses actions requises*. Nous le dénotons par le prédicat FN_a.

Supposons maintenant, en nous inspirant dans l'équation 8.12, que l'agent i n'ait pas inféré $dep\_on_a(i,j,g,i)$ car il ne croit pas que l'agent j puisse entreprendre une action dont il a besoin pour exécuter un plan atteignant g. Si l'agent j peut effectivement entreprendre cette action, c'est lui qui déduit $dep\_on_a(i,j,g,i)$[5]. Autrement dit, une inconsistance au niveau de la société peut être justifiée par

[3] Une fois de plus, nous utilisons dans la suite le terme dépend comme synonyme de a-dépend.

[4] Dans l'annexe A, ces résultats correspondent respectivement au théorème A.2.1 et au corollaire A.2.1.

[5] Rappelons nous que nous supposons que les plans de i et de j sont strictement les mêmes.

le fait qu'un agent puisse avoir des *croyances incomplètes relatives à ses actions requises*. Nous le dénotons par le prédicat IN_a.

Formellement, nous avons :

$$FN_a(i,j,a) \Leftrightarrow \neg \mathbf{B_i}\ is_a(i,a) \wedge \mathbf{B_i}\ is_a(j,a) \wedge \neg \mathbf{B_j}\ is_a(j,a) \quad (8.13)$$

$$IN_a(i,j,a) \Leftrightarrow \neg \mathbf{B_i}\ is_a(i,a) \wedge \neg \mathbf{B_i}\ is_a(j,a) \wedge \mathbf{B_j}\ is_a(j,a) \quad (8.14)$$

Actions proposées

Si nous analysons la situation inverse, quand le même agent i infère le prédicat $dep\_on_a(j,i,g,i)$, son interprétation de cette formule est la suivante, toujours selon l'équation 8.11 :

En utilisant mes plans, je crois que tu dépends de moi pour atteindre le but g car il existe une action nécessaire dans un plan atteignant ce but, que je peux entreprendre et que tu ne peux pas entreprendre.

Appelons cette action une *action proposée*. Supposons maintenant que l'agent j puisse entreprendre cette action. Dans ce cas, nous aurons aussi une inconsistance au niveau de la société, dont la cause est l'existence au sein d'un agent de *croyances incomplètes relatives à ses actions proposées*. Nous le dénotons par le prédicat IO_a.

Supposons maintenant, en nous inspirant de nouveau dans l'équation 8.12, que l'agent i n'ait pas inféré $dep\_on_a(j,i,g,i)$ car il croit que l'agent j peut aussi entreprendre, comme lui, une certaine action nécessaire pour exécuter un plan atteignant g. Si l'agent j ne peut pas entreprendre cette action, c'est lui qui déduit $dep\_on_a(j,i,g,i)$[6]. Autrement dit, une inconsistance au niveau société peut être justifiée par le fait qu'un agent puisse avoir des *croyances fausses relatives à ses actions proposées*. Nous le dénotons par le prédicat FO_a.

Formellement, nous avons :

$$FO_a(i,j,a) \Leftrightarrow \mathbf{B_i}\ is_a(i,a) \wedge \mathbf{B_i}\ is_a(j,a) \wedge \neg \mathbf{B_j}\ is_a(j,a) \quad (8.15)$$

$$IO_a(i,j,a) \Leftrightarrow \mathbf{B_i}\ is_a(i,a) \wedge \neg \mathbf{B_i}\ is_a(j,a) \wedge \mathbf{B_j}\ is_a(j,a) \quad (8.16)$$

Dans la figure 8.2, nous présentons schématiquement ces résultats. Les actions dont l'agent i a besoin sont représentées par des cercles, tandis que les actions qu'il propose sont représentées par des triangles. Nous avons aussi représenté de façon simplifiée la description externe des agents, en ne représentant que les actions qui nous intéressent dans notre analyse.

Quelques exemples

Dans cette section, nous présentons quelques exemples pour illustrer le genre de résultats que nous voulons obtenir. Supposons deux agents ag_1 and ag_2 ayant les descriptions externes présentées dans l'exemple 8.1.

[6]Une fois de plus, nous supposons que les plans de i et de j sont strictement les mêmes.

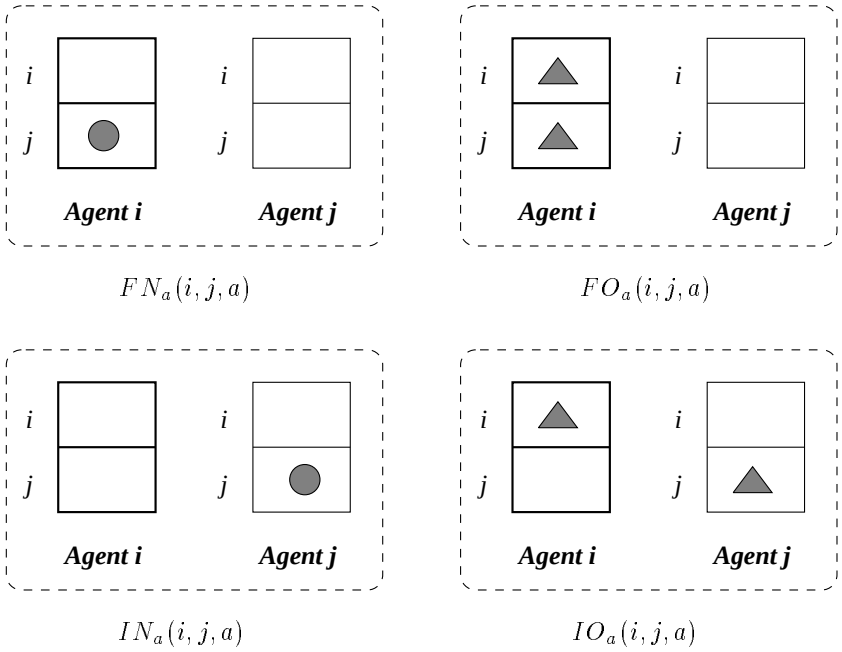


Figure 8.2 : Croyances incomplètes et croyances fausses

Exemple 8.1 : Premier exemple d'inconsistance.

Agent	Buts	Actions	Plans
ag_1	g_1	a_1	$g_1 := a_1(), a_2().$
ag_2	g_1	a_2	$g_1 := a_1(), a_2().$

Description externe de l'agent ag_1

Agent	Buts	Actions	Plans
ag_1	g_1	—	$g_1 := a_1(), a_2().$
ag_2	g_1	a_2	$g_1 := a_1(), a_2().$

Description externe de l'agent ag_2

Clairement, l'agent ag_1 infère une *MBMD* envers l'agent ag_2 pour le but g_1, tandis que ag_2 infère *IND*. Dans cet exemple, l'agent ag_2 a une *croyance incomplète relative à son action requise* a_1. Il ne croît pas que l'agent ag_1 puisse entreprendre cette action. C'est pour cette raison qu'il n'a pas inféré également une *MBMD*.

Supposons maintenant que les agents ag_1 et ag_2 aient les descriptions externes présentées dans l'exemple 8.2.

Exemple 8.2 : Deuxième exemple d'inconsistance.

Agent	Buts	Actions	Plans
ag_1	g_1	a_1	$g_1 := a_1(), a_2().$
ag_2	g_1	a_2	$g_1 := a_1(), a_2().$

Description externe de l'agent ag_1

Agent	Buts	Actions	Plans
ag_1	g_1	a_1	$g_1 := a_1(), a_2().$
ag_2	g_1	—	$g_1 := a_1(), a_2().$

Description externe de l'agent ag_2

Une fois de plus, l'agent ag_1 infère une *MBMD* envers l'agent ag_2 pour le but g_1. Mais cette fois-ci, l'agent ag_2 infère une *UD* envers l'agent ag_1. Comme l'agent ag_2 ne sait pas entreprendre l'action a_2, il n'a rien à offrir à ag_1 en ce qui concerne l'exécution du plan atteignant g_1. Dans cet exemple, l'agent ag_1 a une *croyance fausse relative à son action requise* a_2, car il croit que l'agent ag_2 peut entreprendre cette action, ce qui n'est pas vrai.

Finalement, analysons le cas où nos deux agents ont les descriptions externes présentées dans l'exemple 8.3.

Exemple 8.3 : Troisième exemple d'inconsistance.

Agent	Buts	Actions	Plans
ag_1	g_1	a_1	$g_1 := a_1(), a_2().$
	g_2	—	$g_2 := a_1(), a_3().$
ag_2	g_1	a_2	$g_1 := a_1(), a_2().$
	g_2	a_3	$g_2 := a_1(), a_3().$

Description externe de l'agent ag_1

Agent	Buts	Actions	Plans
ag_1	g_1	a_1	$g_1 := a_1(), a_2().$
	g_2	a_2	$g_2 := a_1(), a_3().$
ag_2	g_1	a_2	$g_1 := a_1(), a_2().$
	g_2	a_3	$g_2 := a_1(), a_3().$

Description externe de l'agent ag_2

Une fois de plus, l'agent ag_1 infère une *MBMD* envers l'agent ag_2 pour le but g_1, mais cette fois-ci l'agent ag_2 infère une *MBRD* envers l'agent ag_1, concernant

les buts g_1 et g_2. Comme l'agent ag_2 croit que l'agent ag_1 peut entreprendre l'action a_2, il n'a rien à offrir à ag_1 en ce qui concerne l'exécution du plan pour atteindre g_1. Par contre, il croit que l'agent ag_1 dépend de lui pour le but g_2 (à cause de l'action a_3). Dans cet exemple, c'est l'agent ag_2 qui a une *croyance fausse relative à son action proposée* a_2, car il croit que l'agent ag_1 peut entreprendre cette action, ce qui n'est pas vrai.

8.3.3 Analyse détaillée des résultats couplés

Nous appelons *résultat couplé* une paire de résultats obtenus par les mécanismes de raisonnement social de deux agents différents, en ce qui concerne leurs situations de but et de dépendance. Nous avons testé tous les résultats couplés possibles. Pour chacun d'entre eux, nous avons proposé et analysé plusieurs exemples afin de distinguer au sein des croyances incomplètes/fausses celles qui sont présentes dans tous les exemples du résultat couplé de celles n'apparaissant que dans un exemple particulier. Pour chaque résultat couplé, nous avons ensuite retenu l'intersection des croyances qui apparaissaient dans tous les exemples, et qui semblaient pourtant être liées au résultat couplé en question. Les résultats de cette analyse sont présentés dans la table 8.2. Dans cette table, nous présentons tous les résultats couplés possibles des mécanismes de raisonnement social de deux agents ag_1 and ag_2 en ce qui concerne le but g_1 (dans les cas où un agent a inféré une dépendance réciproque, nous dénotons par g_2 l'autre but en question). Nous dénotons par FN^i, IN^i, FO^i et IO^i respectivement les prédicats présentés dans les équations 8.13, 8.14, 8.15 et 8.16 introduites auparavant. Nous représentons aussi par FG^i le fait que l'agent i a une *croyance fausse relative à un but de l'autre agent*, dont la représentation formelle complète serait :

$$FG(i,j,g) \quad \Leftrightarrow \quad \mathbf{B_i}\ is_g(j,g) \wedge \neg \mathbf{B_j}\ is_g(j,g) \tag{8.17}$$

Nous voulons discuter les résultats suivants :

1. Dans les cases (3,5), (4,6), (5,3) et (6,4), des *croyances incomplètes relatives aux actions requises* existent. Ce résultat est très intéressant selon une perspective d'analyse du pouvoir social des agents. En effet, prenons les cases (5,3)/(6,4) : l'agent ag_1 ne croyant pas qu'il dépend de l'agent ag_2 pour les buts g_1/g_2, il n'essaiera jamais de former une coalition avec celui-ci qui, également ne le fera pas, puisqu'il est autonome pour le but en question. Nous avons ici une situation où l'agent ag_2 a du *pouvoir social sur* l'agent ag_1 et il est possible qu'aucun des deux ne détecte jamais cette situation. La même conclusion est valable pour les cases (3,5)/(4,6) où l'agent ag_1 a du *pouvoir social sur* l'agent ag_2 ;

2. Un deuxième résultat intéressant est observé dans les cases (5,8), (6,9), (8,5) et (9,6). Dans ces cas, soit l'agent ayant détecté *IND* n'est pas au courant qu'il dépend de l'autre, comme dans le cas précédent, soit celui ayant inféré

une *MBMD/MBRD* a une croyance fausse relative à une action proposée (il croit que l'autre dépend de lui, ce qui n'est pas vrai). Ce cas, contrairement au précédent, peut être éventuellement détecté si l'agent qui a inféré une *MBMD/MBRD* a l'initiative de former une coalition afin d'atteindre son propre but ;

3. Une situation similaire est représentée dans les cases (7,8), (7,9), (8,9), (8,7), (9,7) et (9,8). Dans ce cas, soit l'agent ayant détecté une *UD* (ou une *MBRD* dans les cases (8,9) et (9,8)) a une croyance fausse relative à une action proposée (il croit que l'autre ne dépend pas de lui, ce qui n'est pas vrai), soit l'autre agent a une croyance fausse relative à une action requise (il croit qu'il dépend de l'autre, ce qui n'est pas vrai) ;

4. En analysant les cases (7,7) et (9,9), nous notons que la formule ϕ n'est pas triviale. Ceci provient de l'existence de deux propositions inconsistantes au niveau de la société : $dep\_on(i,j,g,i)$ et $dep\_on(j,i,g,i)$[7]. Chaque agent infère une de ces formules mais pas l'autre. Nous avons ainsi quatre possibilités différentes pour leur valeur de vérité ;

5. Dans les cases (1,8), (2,9), (8,1) et (9,2), l'agent ayant inféré soit une *MBMD* soit une *MBRD* croit que l'autre a le but g_1/g_2 dans son ensemble de buts, ce qui n'est pas vrai ;

6. Dans les cases (3,8), (4,9), (8,3) et (9,4), l'agent qui a inféré soit une *MBMD* soit une *MBRD* a une croyance incomplète relative à ses actions proposées ;

7. Un point intéressant à remarquer est l'existence d'une inconsistance au niveau de la société dans le résultat couplé *UD/UD*, présenté dans la case (7,7). Ce résultat était absent de la table 8.1. Ceci s'explique par notre hypothèse : *les agent ont les mêmes plans et sont conscients de cela.* Dans le cas général, représenté dans la table 8.1, deux agents peuvent avoir des plans différents, et par conséquent, peuvent avoir des entrées de description externe compatibles même s'ils infèrent une *UD* l'un envers l'autre ;

8. La remarque précédente n'est, cependant, pas tout à fait vraie en ce qui concerne la case (9,9). Nous insistons sur le fait que le cas où deux agents infèrent une *MBRD* l'un envers l'autre, présenté dans la table 8.1, ne correspond pas à celui proposé dans la case (9,9) de la table 8.2 : dans celui-ci, l'agent ag_2 infère $MBRD(ag_2, ag_1, g_1, g_2)$ et pas $MBRD(ag_2, ag_1, g_2, g_1)$, qui aurait été tout à fait consistant. La situation présentée dans la table 8.2 correspond effectivement à un cas d'inconsistance au niveau de la société.

[7]Comme les agents ont les mêmes plans, l'agent qui tient lieu d'agent source dans ces formules n'a pas beaucoup d'importance.

G-SIT/D-SIT		ag_2	NG		AUT		DEP				
							IND		UD	$MBMD$	$MBRD$
ag_1			g_1 (1)	g_2 (2)	g_1 (3)	g_2 (4)	g_1 (5)	g_2 (6)	g_1 (7)	g_1 (8)	g_1, g_2 (9)
NG		g_1 (1)								FG^2	
		g_2 (2)									FG^2
AUT		g_1 (3)					IN^2			IO^2	
		g_2 (4)						IN^2			IO^2
DEP	IND	g_1 (5)			IN^1					$IN^1 \vee IO^2$	
		g_2 (6)				IN^1					$IN^1 \vee IO^2$
	UD	g_1 (7)							ϕ	$FO^1 \vee FN^2$	$FO^1 \vee FN^2$
	$MBMD$	g_1 (8)	FG^1		IO^1		$IO^1 \vee IN^2$		$FN^1 \vee FO^2$		$FN^1 \vee FO^2$
	$MBRD$	g_1, g_2 (9)		FG^1		IO^1		$IO^1 \vee IN^2$	$FN^1 \vee FO^2$	$FO^1 \vee FN^2$	ϕ

où $\phi = (FO^1 \wedge FO^2) \vee (FO^1 \wedge FN^1) \vee (FO^2 \wedge FN^2) \vee (FN^1 \wedge FN^2)$

Table 8.2 : Analyse détaillée des résultats couplés

8.4 Révision de la représentation des autres

Les résultats de la section précédente sont encourageants quant à la construction d'un modèle où *un agent peut détecter et raisonner lui-même sur l'inconsistance au niveau de la société*. Même si les agents n'ont pas accès à la description externe des autres agents, puisque ces dernières sont des structures de données *privées*, nous ne pouvons pas oublier la raison pour laquelle nous avons conçu le mécanisme de raisonnement social : *pour permettre aux agents d'entamer des actions sociales*. Cela signifie qu'à chaque fois qu'un agent aura besoin de l'aide des autres, il se servira de ce mécanisme pour *envoyer des propositions de formation de coalition*. Lorsque le récepteur d'un tel message comparera le message envoyé avec la situation de but ou de dépendance qu'il a calculée en utilisant son propre mécanisme de raisonnement social, il pourra utiliser les théorèmes 8.7, 8.8, 8.9 et 8.10 pour détecter et raisonner sur une inconsistance au niveau de la société.

Comme illustration, prenons l'exemple 8.2 et associons les divers buts et actions à ceux introduits dans l'exemple du laboratoire de recherche présenté dans le chapitre 6. Supposons que l'agent $ag3$ vienne d'arriver dans ce laboratoire, et que l'agent $ag2$ ait appris que les papiers soumis à la conférence doivent être écrits en LaTeX : son premier plan n'est plus valable. Par rapport à l'exemple 8.2, nous avons :

- g_1 est le but *write_ss_paper* correspondant à l'écriture de l'article sur la simulation sociale ;
- a_1 correspond à l'action *write_ss_section* ;
- a_2 correspond à l'action *process_latex*.

Supposons que l'agent $ag2$ envoie le message suivant à l'agent $ag3$:

Je crois que nous sommes en MBMD pour le but write_ss_paper. Je te propose une coopération, je me charge d'entreprendre l'action write_ss_section, OK ?

A la réception de ce message, l'agent $ag3$ confirme que l'agent $ag2$ peut entreprendre l'action d'écrire la section principale du papier. Par ailleurs, son mécanisme de raisonnement social établirait une *UD*, ce qui l'amène, en utilisant le théorème 8.7, à conclure de l'existence d'une inconsistance au niveau de la société. En particulier, il peut détecter que la formule $FN^1 \vee FO^2$ est valable (cf. situation représentée dans la case (8,7) de la table 8.2). L'agent $ag2$ lui ayant proposé d'écrire la section principale du papier, il peut inférer que FN^1 est vrai, c'est-à-dire, que l'agent $ag2$ a une croyance erronée en ce qui concerne l'action *process_latex* : cette dernière croit qu'il peut entreprendre cette action, ce qui n'est pas vrai.

Supposons que l'agent $ag3$ réponde à l'agent $ag2$ le message suivant :

Je ne crois pas que je puisse t'aider, car je ne sais pas utiliser LaTeX !

En recevant un tel message, l'agent $ag2$ a besoin d'activer une procédure de *révision de croyances*. Nous justifions ainsi l'affirmation faite dans la section 4.7 : les croyances d'un agent sur d'autres agents l'amènent à une action sociale, et une action sociale de l'autre part peut l'amener à une révision de ces croyances, dans le cas où celle-ci n'est pas réussie. Nous analysons cette révision dans la suite.

8.4.1 Révision de croyances dans un contexte mono-agent

La révision de croyances est un domaine de recherche très répandu, qui a été traité par différentes communautés, soit en tant qu'objet principal de recherche, comme par exemple dans la logique [MS88] ou dans la révision de théories [Bre90], soit en tant que composante importante des systèmes de fusion de données [CD93], de mise à jour de bases de connaissances [ER87] et de bases de données [Cho93, Ost87].

Evidemment, il est hors de propos dans ce travail de proposer un cadre général pour la révision de croyances dans un SMA. Tout ce que nous voulons c'est trouver une solution très simple qui nous permet d'illustrer l'effet du raisonnement social sur la révision de croyances.

Comme [MS88, Dra93], nous considérons qu'un processus de révision de croyances est composé de plusieurs étapes :

1. *détection* : détecter qu'il existe une inconsistance au sein du système ;
2. *identification* : chercher les causes de cette inconsistance ;
3. *décision* : choisir un contexte (un sous-ensemble de croyances consistantes) à maintenir ;
4. *propagation* : retirer les croyances qui seraient à éliminer dans le contexte choisi et éventuellement en déduire d'autres à partir de ce contexte.

Les systèmes de maintien de vérité (TMS) [Doy79, dK86, MS88] abordent rarement le troisième point cité ci-dessus, c'est-à-dire, le choix du contexte à maintenir. Par exemple, dans [Doy79] ce choix est fait de façon aléatoire, et dans [MS88] la question est posée à l'utilisateur du système. Une idée intéressante est présentée dans [Ost87], où on applique un principe général appelé principe de *changement minimal* : on choisit de retirer la cause dont l'élimination a comme conséquence le plus petit changement possible dans la base de croyances courante. D'ailleurs, ce principe, parmi d'autres, a été aussi utilisé dans le cas des SMA dans [Gal88].

Aucun des moyens présentés ci-dessus pour le choix du contexte à maintenir ne nous convient, la non-pertinence des deux premiers étant évidente. En ce qui concerne le principe du changement minimal, il présente quelques inconvénients lors de son application dans un contexte multi-agents, car la source d'information

n'y est pas représentée[8]. Nous croyons que cette notion est importante, comme nous le montrons dans la suite.

Dans notre contexte, les étapes de détection et d'identification sont faites par comparaison entre les résultas couplés des mécanismes de raisonnement social de deux agents, en particulier en utilisant les théorèmes 8.7, 8.8, 8.9 et 8.10. Particulièrement, l'identification peut être faite en utilisant la table 8.2, comme nous venons de le montrer dans l'exemple du laboratoire de recherche. En ce qui concerne la propagation, tout ce qu'il y a à faire est une expansion (ajout d'un élément) ou une contraction (retrait d'un élément) des prédicats présentés dans la section 5.3.2, qui représentent la description externe. Les diverses propriétés du modèle, telles que les relations de dépendance, doivent aussi être mises à jour, mais cette procédure n'est pas très compliquée.

8.4.2 Révision de croyances dans un contexte multi-agents

Plusieurs travaux ont été réalisés pendant les dernières années concernant des modèles théoriques pour la révision de croyances dans un contexte multi-agents [Gal91, Gas91, Dra93] et des modèles et implémentations pour les systèmes de maintien de la vérité distribués (DTMS) [MJ89, HB91, MJO94].

Par exemple, dans [MJ89, HB91], comme les agents sont bienveillants par hypothèse, à chaque fois qu'un agent reçoit une information d'un agent qui en est *responsable*, il l'incorpore automatiquement. La définition de responsabilité d'un agent pour une information est généralement définie lors de la phase de conception du système. Ainsi, ces modèles ne nous conviennent pas, car ils ne peuvent pas être appliqués à un SMA ouvert. Dans [MJO94], cette notion de responsabilité sur une information n'est pas exploitée de manière très rigide. Par contre, lorsqu'un agent détecte une contradiction entre une proposition qui est à la fois justifiée en interne (par exemple l'agent croit en ϕ) et à la fois justifiée externement (par exemple, un autre agent lui communique qu'il ne croit pas en ϕ), la décision de choisir le contexte est laissée à l'utilisateur du système.

Une approche intéressante est présentée dans [Dra93]. Dans ce modèle, on représente la *crédibilité des divers sources d'information* qu'un agent peut avoir : le raisonnement, la perception et la communication. La crédibilité interne d'une formule est calculée comme étant le produit de la crédibilité de la source par la crédibilité externe de la formule, qui est aussi fournie lors de la communication. En utilisant une approche quantitative, ce modèle nous permet de représenter des inconsistances : la crédibilité des formules est calculée séparément de celle de leurs négations.

Une deuxième approche très intéressante est présentée dans [Gas94]. Dans ce travail, un agent représente les autres en spécifiant leur crédibilité et leur autorité par rapport à des *thèmes d'information*. Ainsi, lorsqu'un agent reçoit un message d'un autre, il utilise ces derniers aspects pour décider quel contexte maintenir.

[8] Dans les systèmes mono-agent, la notion de source d'information est absente.

Le modèle de sous-théories préférées [Bre90] est utilisé comme une base théorique pour cette approche. Une approche qui utilise aussi l'idée de thèmes dans un contexte de fusion de bases de données est présentée dans [Cho94].

Nous avons retenu quelques idées de ces deux travaux pour concevoir notre critère de choix : nous différencions explicitement la perception de l'inférence en tant que sources d'information sur les autres, comme dans [Dra93], et nous adoptons aussi la notion de thème d'information, d'une façon très restreinte, comme nous le montrons dans la suite.

8.5 Choix de contexte

Dans les sections suivantes, nous présentons informellement la procédure de choix de contexte, nous proposons un modèle pour la caractériser et nous montrons comment le modèle d'agent introduit dans la section 3.4 est capable de la gérer.

8.5.1 Procédure de choix de contexte

En ce moment, il faut rappeler deux principes introduits dans la section 1.3 : le principe de la sincérité (P2) et le principe de l'auto-connaissance (P3). En utilisant ces principes, nous avons un résultat très intéressant :

Si un agent i a une connaissance complète et correcte relative à lui-même et s'il ne communique toujours que ce qu'il croît, alors pour tous les agents, la meilleure source d'information possible sur l'agent i est l'agent i lui-même.

Nous devons ainsi proposer un critère de décision du contexte à maintenir basé sur cette propriété. Comme nous l'avons déjà dit auparavant, nous ne nous intéressons pas à proposer un critère complet et général, nous voulons seulement en proposer un qui nous paraisse bien fondé et qui puisse être appliqué de façon très simple. Pour cela, nous adoptons tout d'abord que (i) la perception est une source d'information moins crédible que l'inférence et (ii) un agent considère ses propres sources d'information (perception et inférence) plus crédibles que les informations envoyés par des autres agents, à l'exception de l'agent auquel l'information fait référence, comme nous venons de le présenter ci-dessus.

8.5.2 Critère de choix de contexte

Soit $SR = (i, IL, A, S, topic, source, \prec_{context}, decide_{context})$ un modèle pour caractériser le comportement d'un agent en ce qui concerne son choix de contexte, tel que :

- i désigne l'agent à qui ce modèle appartient ;
- IL désigne le sous-ensemble du langage interne de l'agent i dont les formules décrivent les propriétés des agents ;

- A désigne un ensemble d'agents ;
- $S = \{I, P\} \cup A$ est un ensemble de *sources d'information*, tel que nous l'avons défini dans la section 5.1.2 où :
  - $\star$ $j \in A$ désigne un message reçu de l'agent j ;
  - $\star$ P désigne la perception ;
  - $\star$ I désigne l'inférence ;
- $topic : IL \mapsto A$ est une fonction qui donne pour chaque formule $\phi \in IL$ l'identification de l'agent auquel la formule fait référence ;
- $source : IL \mapsto S$ une fonction qui donne pour chaque formule $\phi \in IL$ la source de cette formule. Pour une question d'uniformisation de notation, nous supposons que la connaissance que i a sur lui-même est représentée par la source I, c'est-à-dire, pour toute formule $\phi \in IL$ telle que $topic(\phi) = i$, nous avons $source(\phi) = I$.
- $\prec_{context}$ est la relation d'ordre partielle définie dans la suite ;
- $decide_{context} : IL \times IL \mapsto IL$ est une fonction qui, à partir de deux formules inconsistantes ϕ et ϕ' appartenant à IL, en choisit une comme le contexte à maintenir, cette fonction étant définie dans la suite.

Montrons maintenant comment nous construisons la relation d'ordre partielle $\prec_{context}$. Notons $\prec'_{context}$ la relation sur IL^2 suivante :

Définition 8.5.1. *Si* $\phi, \phi' \in IL, j, k \in A, j \neq k$ *et* $topic(\phi) = topic(\phi') = j$ *alors* $\phi' \prec'_{context} \phi$ *ssi :*

- $source(\phi') = k$ *et* $source(\phi) = P$ *ou*
- $source(\phi') = P$ *et* $source(\phi) = I$ *ou*
- $source(\phi') = I$ *et* $source(\phi) = j$.

Soit $\prec_{context}$ la fermeture transitive de $\prec'_{context}$. La preuve du lemme suivant est évidente, puisque la relation $\prec'_{context}$ est anti-symétrique :

Lemme 8.5.2. $(IL, \prec_{context})$ *est un ensemble partiellement ordonné.*

Ce critère est représenté sous forme de diagramme dans la figure 8.3.

Maintenant, nous pouvons définir la fonction de choix de contexte $decide_{context}$ de la façon suivante :

$$decide_{context}(\phi, \phi') = \begin{cases} \phi & \text{si } \phi' \prec_{context} \phi \\ \phi' & \text{si } \phi \prec_{context} \phi' \\ \phi' & \text{si } \phi \not\prec_{context} \phi' \text{ et } \phi' \not\prec_{context} \phi \end{cases}$$

Selon ce critère, et par rapport à sa représentation de l'agent j, un agent i préférera toujours croire aux informations fournies par j sur lui-même. En revanche, il va toujours éviter de croire dans les informations sur j fournies par les autres agents. Si les deux sources d'information ne sont pas comparables, alors l'agent garde le contexte le plus récent.

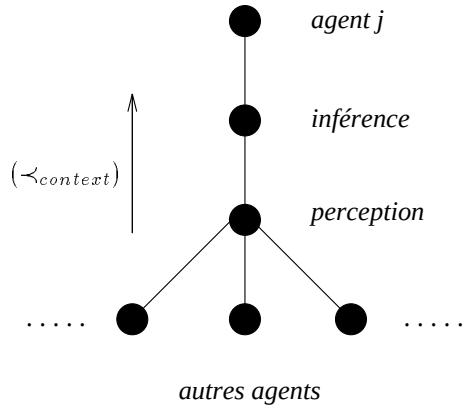


Figure 8.3 : Critère de choix de contexte

8.5.3 Gestion de choix de contexte par le modèle d'agent

En instanciant le critère présenté ci-dessus sur notre modèle d'agent, nous avons :

- IL est le sous-ensemble de formules de notre langage interne correspondant aux formules d'interfaçage avec la description externe, présentées dans la section 5.3.2 ;
- la fonction *topic* retourne le premier terme de chaque prédicat de IL : par exemple, nous avons $topic(is_g(j, g)) = j$;
- la fonction *source* correspond aux composantes $s(g), s(a), s(r)$ et $s(p)$ des équations 5.4, 5.6, 5.8 et 5.10 ;

La gestion de choix de contexte par le modèle d'agent est présentée dans la figure 8.4. En utilisant la description externe (RS) et les résultats présentés dans la section 8.3, le mécanisme de raisonnement calcule les propositions pouvant être

inconsistantes et les envoie au mécanisme de décision (DS). Celui-ci, en utilisant le critère de choix de contexte $\prec_{context}$ choisit un contexte à maintenir (CS), et le mécanisme de révision met à jour l'entrée correspondante de la description externe et révise les propositions inférées précédemment. Dans le cas où l'origine de l'inconsistance a été une proposition de coalition basée sur une fausse croyance du proposant, un message de révision lui est envoyé.

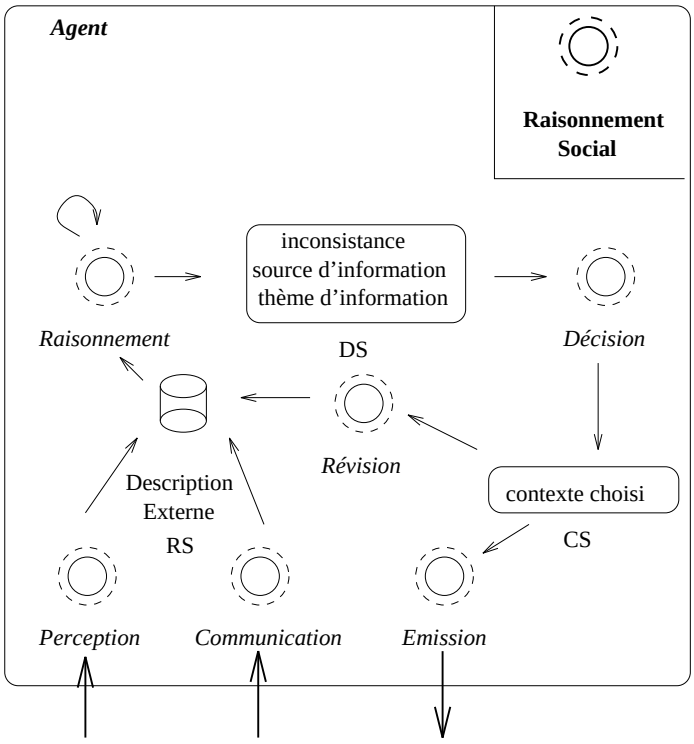


Figure 8.4 : Gestion de choix de contexte par le modèle d'agent

8.6 Discussion

Dans ce chapitre, nous avons montré que notre mécanisme de raisonnement social permet à un agent de détecter que sa représentation des autres agents est incorrecte ou incomplète. Nous avons aussi indiqué que les approches basées sur les DTMS ne peuvent pas être utilisées dans notre contexte de systèmes ouverts, car dans ceux-ci l'attribution de la responsabilité des agents par rapport à certaines classes de propositions ne peut pas être faite pendant la phase de conception du système. Nous avons ainsi proposé une alternative à ces méthodes. En prenant comme base les principes de l'auto-connaissance (P3) et de la sincérité (P2), nous avons conçu un critère de choix de contexte qui utilise les notions de source et de thème d'information. Nous avons aussi montré que notre modèle d'agent est

capable de gérer de tels choix.

La méthode de manutention de consistance proposée dans ce chapitre est *incrémentale* : elle est activée lorsque les agents interagissent les uns avec les autres, en essayant de former des coalitions pour atteindre leurs buts. De cette manière, nous justifions l'affirmation faite dans la section 4.7 concernant la relation bidirectionale, de cause à effet, entre l'action sociale et la révision de croyances : chacune d'entre elles est à la fois la cause et à la fois l'effet de l'autre. Un autre aspect intéressant est que cette méthode ne présupose pas que les agents auront toujours des croyances correctes et complètes les uns des autres, situation fortement improbable dans les systèmes auxquels nous nous intéressons.

Troisième partie

Implémentation et applications

Chapitre 9

Implémentation du modèle

Dans ce chapitre, nous décrivons l'implémentation de notre modèle, pour laquelle nous avons utilisé une programmation orientée objet. Une première version de cette implémentation a été initialement présenté dans [SD94b]. Tout au long de ce chapitre, nous utilisons un exemple du monde des blocs, représenté dans la figure 9.1, où nous représentons une société composée de quatre agents : $ag1$, $ag2$, $ag3$ et $ag4$.

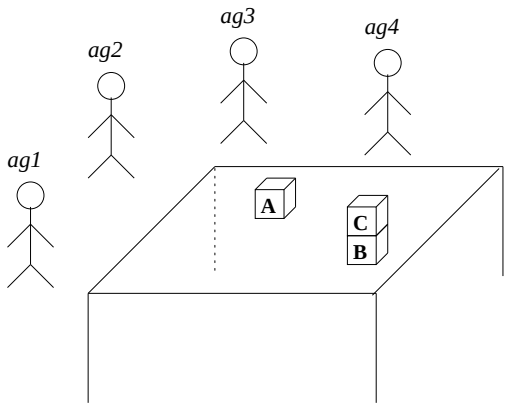


Figure 9.1 : Deuxième exemple du monde des blocs

Ce chapitre est composé de quatre sections. Dans la section 9.1, nous décrivons les classes conçues pour implémenter la description externe et nous illustrons son utilisation par un exemple d'insertion d'information concernant un agent. Dans la suite, nous introduisons dans la section 9.2 la notion de *réseau de dépendance* et nous montrons les classes qui ont été définies pour son implémentation. Enfin, dans les sections 9.3 et 9.4 nous décrivons respectivement les algorithmes de calcul pour les situations de buts et les situations de dépendance.

9.1 Description externe

Comme nous l'avons exposé dans la section 5.1, notre mécanisme de raisonnement social utilise une structure de données appelée *description externe* pour conserver les informations qu'un agent possède sur les autres. Chaque agent a sa propre description externe qui lui est *privée*. Celle-ci est composée de plusieurs *entrées*, chacune d'entre elles étant composée des buts, des actions, des ressources et des plans relatifs à un agent particulier.

Dans notre exemple, les buts des agents sont d'empiler des blocs les uns sur les autres, de dépiler des blocs et de les peindre. Les actions disponibles sont empiler (*put_on*), dépiler (*clear*) et peindre (*paint*). Les blocs A, B et C sont considérés comme des ressources. La description externe complète de notre exemple[1] est présentée dans l'exemple 9.1.

L'ensemble de classes utilisées pour l'implémentation de notre modèle de description externe est présenté dans la figure 9.2.

Dans cette figure, nous avons deux types de classes : des classes *primitives* et des classes *conteneurs*. Des classes primitives représentent des concepts primitifs de notre modèle, tels que des buts, des actions, des ressources et des plans. Des classes conteneurs servent à représenter des ensembles de ces concepts. Ainsi, une instance d'une classe conteneur est formé par un ensemble d'instances de sa classe primitive[2].

Exemple 9.1 : Description externe de l'exemple du monde des blocs.

Agent	Buts	Actions	Ressources	Plans
$ag1$	on(C,table)	put_on	B	on(C,table):=clear(C).
$ag2$	on(A,B)	—	C	on(A,B):=clear(C), put_on(A,B). on(A,B):=put_on(A,B).
	color(A,blue)	—	—	color(A,blue):=paint(A,blue).
$ag3$	on(A,B)	clear	A	on(A,B):=clear(C), put_on(A,B).
$ag4$	color(B,red)	clear	—	—

Une description externe ED (équation 5.1) n'étant qu'un ensemble d'entrées correspondant à chaque agent de la société, celle-ci est implémentée par la classe conteneur external_description. Chacune de ses entrées $ED(j)$ (équation 5.2) est implémentée par la classe primitive ext_descr, dont les composantes sont les suivantes :

[1] Nous ne spécifions pas auquel de ces agents cette description appartient, car ce n'est pas important vis-à-vis de l'objectif de ce chapitre, une fois que l'implémentation est commune à tous les agents. Néanmoins, nous voulons insister sur le fait que dans notre modèle, chaque agent a sa propre description externe qui lui est privée.

[2] Cet ensemble pouvant être vide.

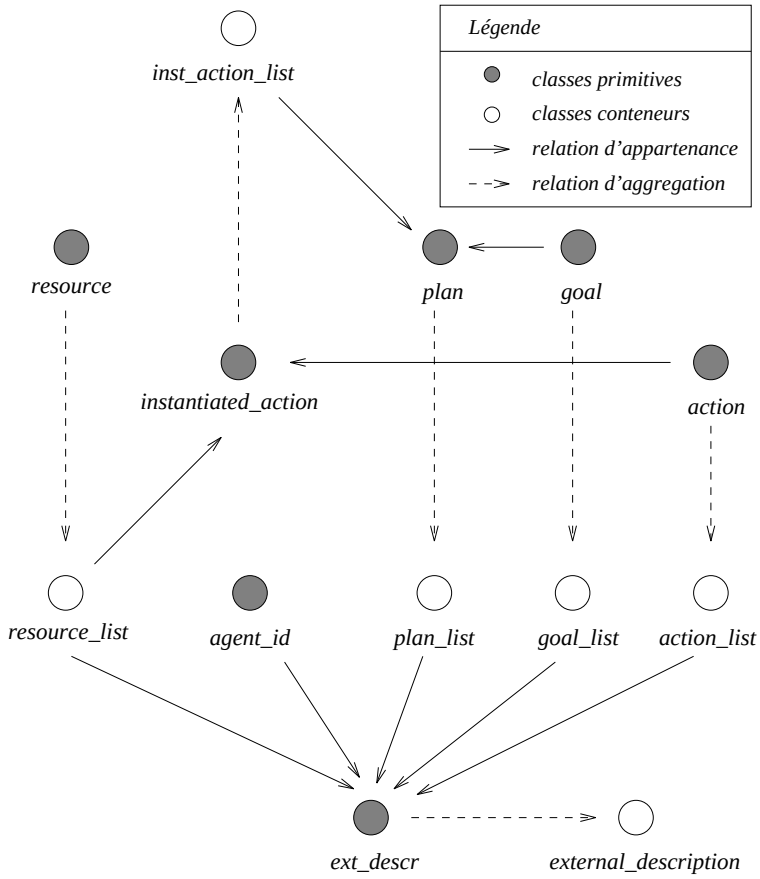


Figure 9.2 : Classes composantes de la description externe

1. la classe primitive agent_id, qui contient l'identification de l'agent dont l'entrée j fait référence. Cette identification est composée de son nom, du nom de la machine où cet agent est exécuté et d'un nombre entier qui identifie le processus associé ;

2. la classe conteneur goal_list, qui contient la liste de buts $G(j)$ (équation 5.3) de l'agent ;

3. la classe conteneur action_list, qui contient la liste d'actions $A(j)$ (équation 5.5) de l'agent ;

4. la classe conteneur resource_list, qui contient la liste de ressources $R(j)$ (équation 5.7) de l'agent ;

5. la classe conteneur plan_list, qui contient la liste de plans $P(j)$ (équation 5.9) de l'agent.

Etant donné un plan p, nous avons besoin de représenter ses *action instanciées* $I(p)$ (équation 5.11), c'est-à-dire une liste d'actions avec la liste de ressources qu'elles utilisent. Dans notre exemple, le plan de l'agent $ag3$ pour atteindre son but $on(A,B)$ est $on(A,B) := clear(C), put\_on(A,B)$. Ce plan est composé de deux actions instanciées : $clear(C)$ et $put\_on(A,B)$. Pour la première, l'action correspondante est *clear* et la ressource est le bloc C. Pour la deuxième, l'action correspondante est *put_on* et les ressources utilisées sont les blocs A et B. De cette façon, un plan p (équation 5.10) est implémenté par la classe plan, dont les composantes sont les suivantes :

1. le nom du but $id_g(p)$ (équation 5.10) que le plan doit atteindre ;
2. la classe conteneur inst_action_list, qui contient la liste d'actions instanciées $I(p)$ (équation 5.10) utilisées par ce plan ;
3. l'identification de la source d'acquisition $s(p)$ (équation 5.10) de cette information (perception, inference ou communication). Dans ce dernier cas, on ajoute l'identification de l'agent qui a envoyé cette information (implémenté par la classe agent_id) ;
4. une valeur entière qui sert à stocker le coût de ce plan. Cette valeur est remplie lorsqu'un agent doit choisir un plan pour atteindre un certain but, comme nous l'avons expliqué dans la section 6.3.

Finalement, chaque action instanciée i (équation 5.12) est implémentée par la classe primitive instantiated_action, dont les composantes sont les suivantes :

1. le nom de l'action $id_a(i)$ (équation 5.12) à entreprendre ;
2. une liste de noms de ressources $R(i)$ (équation 5.12) utilisées par cette action.

Les classes primitives goal, action et resource sont composées des éléments suivants :

1. le nom du but, de l'action ou de la ressource ;
2. une identification de la source d'acquisition de l'information $s(g)$, $s(a)$ ou $s(r)$ (équations 5.4, 5.6 et 5.8) comme dans le cas des plans décrit ci-dessus ;
3. une valeur entière correspondant aux fonctions d'importance des buts $w(g)$ et de coût des actions et des ressources $c(a)$ et $c(r)$, telles que représentées dans les mêmes équations citées ci-dessus.

En considérant notre modèle d'agent présenté dans la section 3.4, un agent est capable de communiquer avec les autres et de percevoir son environnement. En particulier, en utilisant ses capacités, ses croyances sur les autres peuvent *changer*

dynamiquement, et par conséquent il doit être capable de mettre à jour sa description externe. Pour cette raison, toutes les méthodes associées aux classes décrites ci-dessus ont été conçues de façon à pouvoir être activées de façon incrémentale, pour permettre une mise à jour efficace. Dans l'exemple 9.2, nous présentons le code correspondant à l'insertion de l'information concernant l'agent $ag3$.

Exemple 9.2 : Exemple de code pour l'insertion d'un agent.

```
#include "depnet.h"
#include "streamwb.h"
#include "stringwv.h"
#include "slist.h"
#include "glist.h"
#include "exdescr.h"
#include "depnetdef.h"

external_description* ext;
ext_descr* ag;
plan* pl;
instantiated_action* ac;
char host[64];

main()
{
// Get the host name
gethostname(host,64);
// Create external description
ext = new external_description();
// Creates the agent
creates_ag3();
// Prints the external description
ext->print();
}
```

```
creates_ag3 ()
{
// Create agent ag3
ag = new ext_descr ();
// Set agent_id slots
ag->set_name("ag3");
ag->set_host(host);
ag->set_pid(getpid());
// Set goals
ag->append_goal("on(A,B)");
// Set actions
ag->append_action("clear");
// Set resources
ag->append_resource("A");
// Set plans
pl = ag->append_plan_goal ("on(A,B)");
ac = ag->append_plan_action(pl,"clear");
ag->append_plan_resource(ac,"C");
ac = ag->append_plan_action(pl,"put_on");
ag->append_plan_resource(ac,"A");
ag->append_plan_resource(ac,"B");
// Includes in external description
ext->append(ag);
}
```

Nous présentons dans la figure 9.3 les instances des diverses classes décrites ci-dessus qui sont créées par l'exécution du code présenté dans l'exemple 9.2. Nous supposons que l'agent $ag3$ est exécuté sur une machine appelée *cosmos* et dont l'identification de processus est *2035*. Les instances et noms qui apparaissent au dessus de la classe, dans les carrés pointillés, sont celles utilisées pour créer l'instance de la class plan. Celles qui apparaissent au dessous de la classe, dans les carrés avec les coins arrondis, correspondent effectivement aux propriétés de l'agent décrit. Par concision, nous ne présentons ni dans l'exemple 9.2 ni dans la figure 9.3 les informations concernant la source des informations, l'importance de buts et les coûts des actions et des ressources.

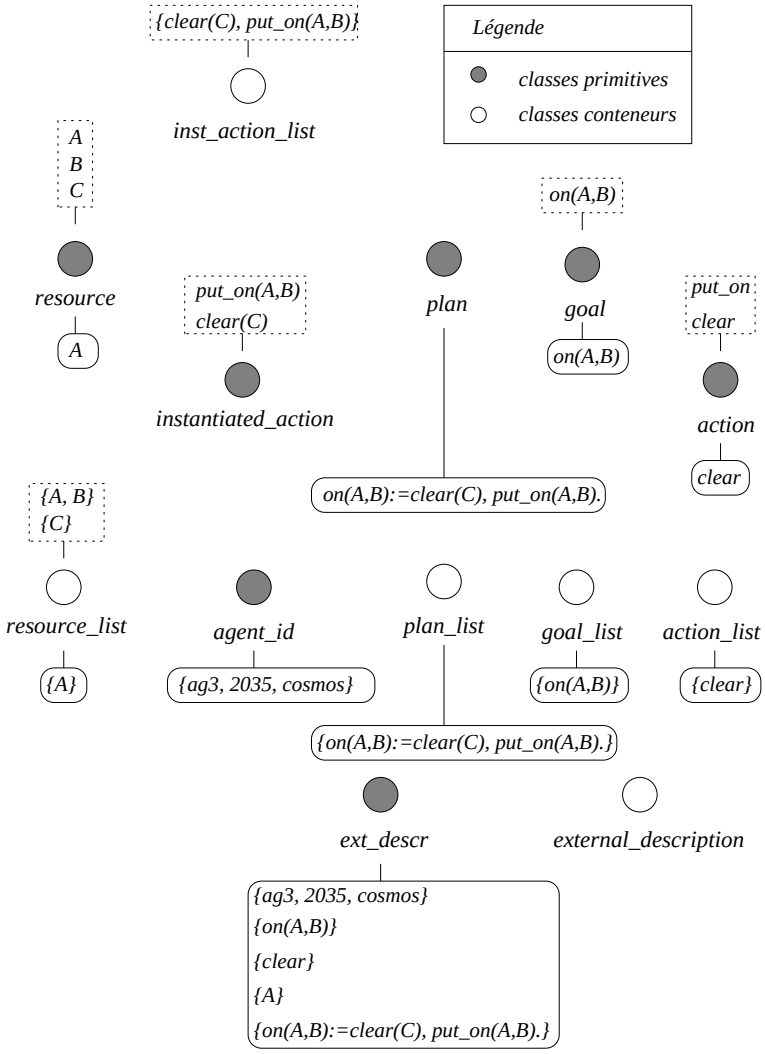


Figure 9.3 : Exemple d'instanciation d'une description externe

9.2 Réseaux de dépendance

Nous avons implémenté une structure de données appelée *réseau de dépendance* pour pouvoir représenter de façon compacte les diverses relations de dépendance d'un certain agent objet envers des autres agents, telles que définies dans la section 5.3.6. Nous appelons respectivement *réseau de a-dépendance/r-dépendance* des structures qui contiennent toutes les a-dépendances/r-dépendances d'un agent. D'ailleurs, dans la section 5.2.1, nous avons aussi insisté sur le fait que notre modèle permet à un agent sujet i de calculer les dépendances d'un agent objet j en utilisant les plans de divers agents source k.

Ainsi, pour construire un réseau dep_net(i, type, j), on doit spécifier les trois paramètres suivants :

- l'agent objet i auquel le réseau se réfère, spécifié par une instance de la classe agent_id ;
- le type du réseau (action ou resource) ;
- l'agent source j duquel l'agent sujet utilise les plans, spécifié aussi par une instance de la classe agent_id.

Selon le modèle présenté dans le chapitre 5, l'implémentation d'une relation de dépendance $dep\_on_a(i, j, g, i)$ (équation 5.37) est réalisée par un *chemin dans le réseau* dep_net(i, action, i). Dans le réseau, le plan utilisé pour déduire la dépendance, ainsi que l'action dont l'agent objet i a besoin, sont représentés explicitement.

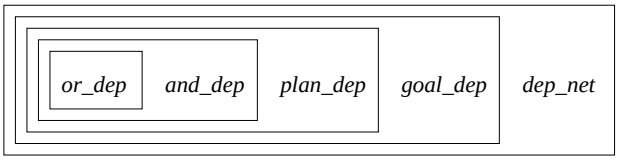


Figure 9.4 : Implémentation inclusive des réseaux de dépendance

L'implémentation d'un réseau est faite par des *inclusions successives*, en exprimant des différents niveaux d'abstraction, selon illustré dans la figure 9.4 et détaillé dans la suite. Pour cela, nous avons exploité les notions de *ou-dépendance* et *et-dépendance*[3], initialement présentés dans la section 4.6.1. Nous avons conçu un ensemble de classes *primitives* : or_dep, and_dep, plan_dep, goal_dep et dep_net. Pour chacune d'entre elles, sauf pour la classe dep_net, nous avons aussi implémenté une classe *conteneur* qui lui est associée, comme le montre la figure 9.5.

Prenons comme exemple l'agent objet $ag2$. Son réseau de a-dépendance, en considérant ses propres plans dep_net(ag2, action, ag2), est implémenté de la façon suivante :

[3]Particulièrement celle concernant la dépendance multi-partite.

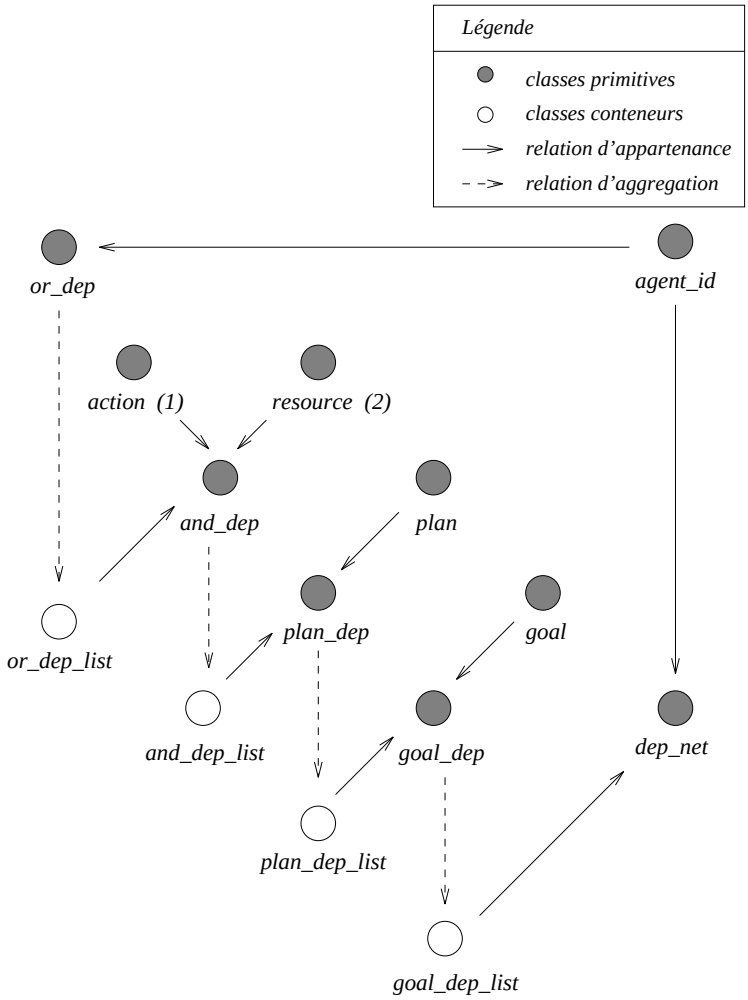


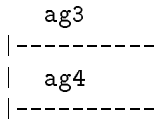
Figure 9.5 : Classes composantes du réseau de dépendance

1. la classe or_dep implémente notre notion moins spécifique de dépendance, l'ou-dépendance. Ses instances ne contiennent que l'identification de l'agent tiers de qui l'agent objet dépend, implémenté par la classe agent_id. Si nous prenons le première plan de l'agent $ag2$ pour le but $on(A, B)$, cet agent est dépendant de l'agent $ag3$ en ce qui concerne l'action $clear$:

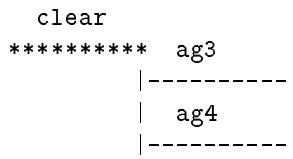
```
                    ag3
                  |----------
```

2. la classe conteneur or_dep_list implémente une liste de ou-dépendances. Dans

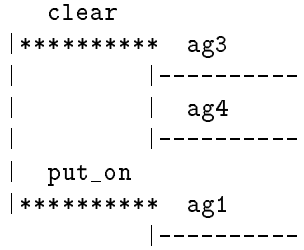
le cas général, plusieurs agents peuvent être capables d'entreprendre une action/contrôler une ressource. C'est pour cette raison que nous avons appelé cette notion ou-dépendance, car il suffit qu'un agent rende cette action/ressource disponible pour que le plan en question soit exécutable. Dans notre exemple, deux agents, $ag3$ et $ag4$, sont capables d'entreprendre l'action *clear* :



3. la classe and_dep implémente notre notion de et-dépendance. Elle est composée de l'action/ressource dont l'agent objet est dépendant (implémentée soit par la classe action soit par la classe ressource) et d'une liste d'ou-dépendances. Si l'action/la ressource en question est non-disponible, tel que nous l'avons introduit dans la section 6.1, cette liste est vide et la méthode print de cette classe fournit la sortie `UNKNOWN`. Dans notre exemple, l'et-dépendance concernant l'action *clear* est :

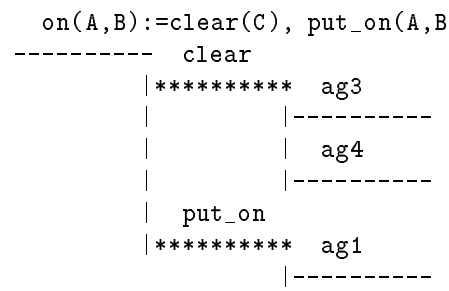


4. la classe conteneur and_dep_list implémente une liste de et-dépendances. Dans le cas général, un agent peut avoir besoin de plus d'une action/ressource pour exécuter un plan. C'est pour cette raison que nous avons appelé cette notion et-dépendance, car il faut que toutes ces actions/ressources soient disponibles pour que le plan en question soit exécutable. Dans notre exemple, comme l'agent $ag2$ n'est pas capable non plus d'entreprendre l'action *put_on*, il est aussi dépendant pour cette action (cette fois-ci de l'agent $ag1$) :

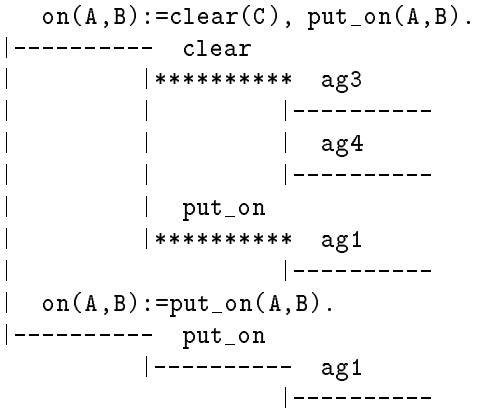


5. la classe plan_dep implémente notre notion de plan-dépendance. Elle est est composée d'un plan (implémentée par la classe plan) et d'une liste de

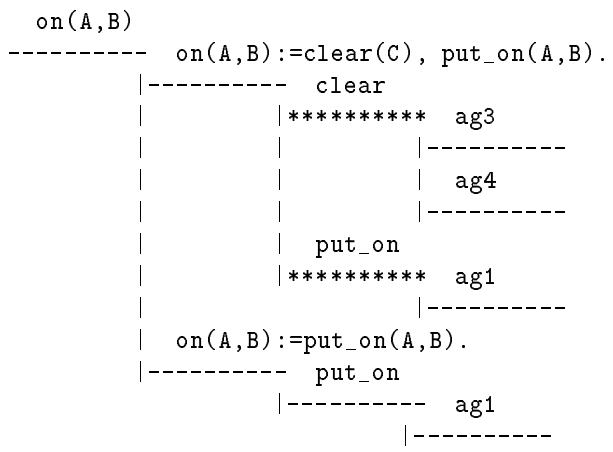
et-dépendances. Si l'agent est a-autonome/r-autonome pour le but auquel le plan se réfère, cette liste est vide et la méthode print de cette classe fournit la sortie `A-AUTONOMOUS/R-AUTONOMOUS`. Dans notre exemple, la plan-dépendance concernant le premier plan de l'agent $ag2$ pour le but $on(A, B)$ est :



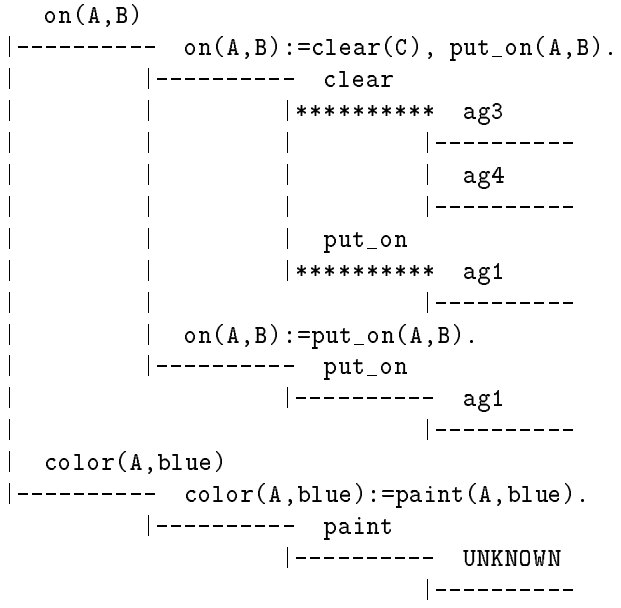
6. la classe conteneur plan_dep_list implémente une liste de plan-dépendances. Dans le cas général, un agent peut avoir plus d'un plan pour atteindre un certain but. En effet, dans notre exemple l'agent $ag2$ a deux plans possibles pour atteindre le but $on(A, B)$:



7. la classe goal_dep implémente notre notion de but-dépendance. Elle est composée d'un but (implémentée par la classe goal) et d'une liste de plan-dépendances. Si l'agent n'a aucun plan pour atteindre le but en question, cette liste est vide et la méthode print de cette classe fournit la sortie `NO-PLANS`. Dans notre exemple, la but-dépendance de l'agent $ag2$ pour le but $on(A, B)$ est :



8. la classe conteneur goal_dep_list implémente une liste de but-dépendances. Dans le cas général, un agent peut avoir plus d'un but à atteindre. En effet, dans notre exemple l'agent $ag2$ a deux buts à atteindre, $on(A, B)$ et $color(A, blue)$:



Selon les définitions introduites dans la section 6.1, l'action *paint* est non-disponible, car aucun agent appartenant à la société peut l'entreprendre. Par conséquent, nous avons respectivement que le plan $color(A, blue) := paint(A, blue)$ est non-exécutable et le but $color(A, blue)$ est non-réalisable ;

9. finalement, la classe dep_net implémente sur une même structure toutes les dépendances d'un agent d'un certain type. Comme nous l'avons déjà

mentionné auparavant, elle est composée de l'identification de l'agent objet auquel le réseau se réfère (spécifié par une instance de la classe agent_id), d'une identification du type du réseau (action ou resource), d'une identification de l'agent source duquel on utilise les plans (spécifié aussi par une instance de la classe ext_descr) et d'une liste de but-dépendances. Si l'agent n'a aucun but à atteindre, cette liste est vide et la méthode print de cette classe fournit la sortie NO-GOALS. Dans notre exemple, le réseau de a-dépendance final de l'agent $ag2$ dep_net(ag2, action, ag2) est :

```
  ag2
 <ag2>
----------  on(A,B)
         |----------  on(A,B):=clear(C), put_on(A,B).
         |          |----------  clear
         |          |          |**********  ag3
         |          |          |          |----------
         |          |          |          |  ag4
         |          |          |          |----------
         |          |          |  put_on
         |          |          |**********  ag1
         |          |                     |----------
         |          |  on(A,B):=put_on(A,B).
         |          |----------  put_on
         |                     |----------  ag1
         |                                |----------
         |  color(A,blue)
         |----------  color(A,blue):=paint(A,blue).
                    |----------  paint
                               |----------  UNKNOWN
                                          |----------
```

Dans notre exemple, le réseau de a-dépendance dep_net(ag3, action, ag3) de l'agent $ag3$ est plus simple :

```
  ag3
 <ag3>
----------  on(A,B)
         |----------  on(A,B):=clear(C), put_on(A,B).
                    |----------  put_on
                               |----------  ag1
                                          |----------
```

Dans la figure 9.6, nous présentons les instances des diverses classes décrites ci-dessus qui sont créées pour construire le réseau de a-dépendance de l'agent $ag3$ dep_net(ag3, action, ag3). Dans les carrés, les listes sont représentées entre accolades, et chacun de ses éléments sont représentés entre crochets.

Comme exemple de méthodes développées pour la classe dep_net, nous pouvons citer :

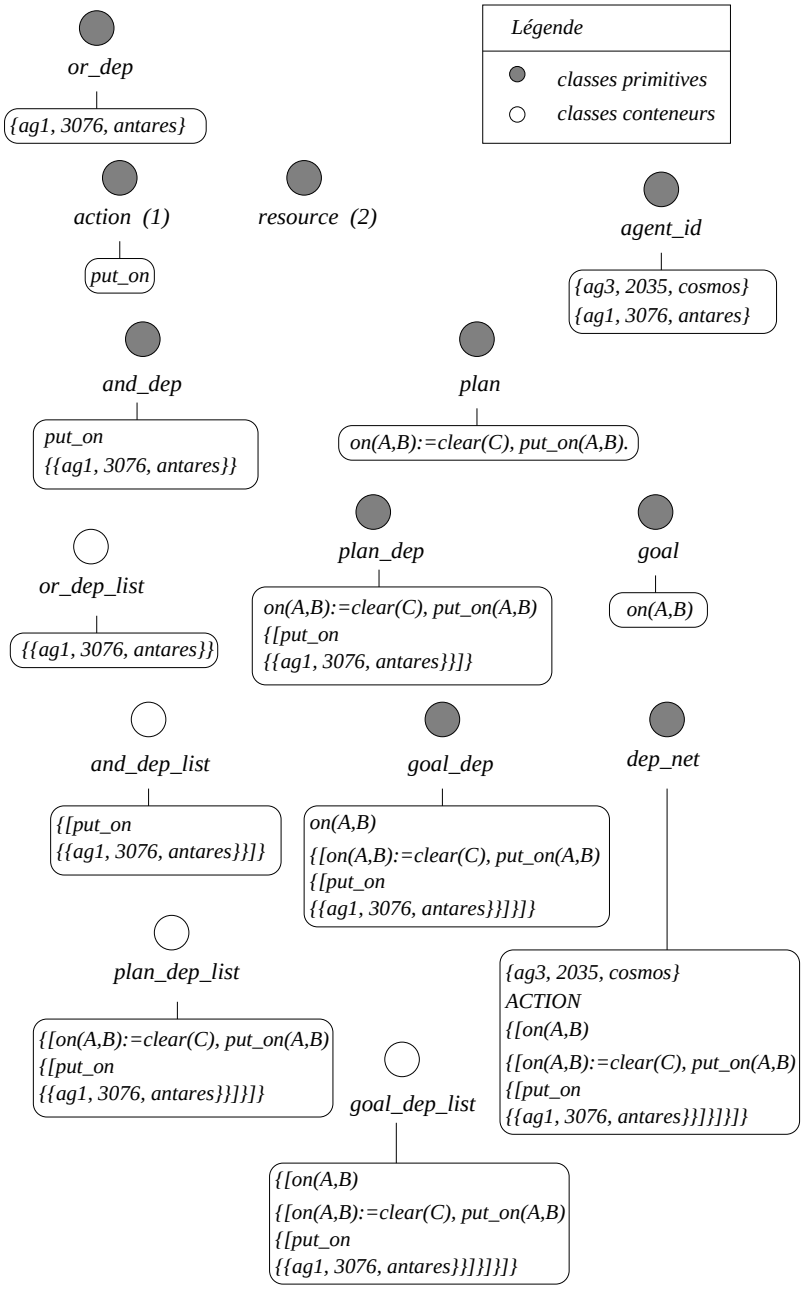


Figure 9.6 : Exemple d'instanciation d'un réseau de dépendance

- set_agent: remplit l'identification de l'agent auquel le réseau se réfère ;
- insert_dep: insère une ou-dépendance dans le réseau ;
- a_autonomous/r_autonomous: vérifie si un agent est a-autonome/r-autonome pour un certain but ;
- depend_a_on_other/depend_r_on_other: vérifie si un agent a-dépend/r-dépend d'un autre pour un certain but ;

Dans le cas d'insertion d'une nouvelle ou-dépendance, une nouvelle instance de la classe goal_dep est automatiquement créée si le but en question n'était pas encore représenté dans le réseau. La même stratégie est utilisée en ce qui concerne un nouveau plan (une nouvelle instance de la classe plan_dep est créée) ou bien une nouvelle action/ressource dont l'agent objet, auquel le réseau se réfère, est dépendant (une nouvelle instance de la classe and_dep est créée).

9.3 Situations de but

En utilisant les éléments de notre modèle décrit dans le chapitre 5, nous présentons dans la figure 9.7 l'algorithme utilisé par un agent sujet i pour calculer sa situation de but relative au but g.

Dans cette figure, nous représentons les divers prédicats de notre modèle qui sont testés pendant ce calcul. Notons que selon le modèle, l'agent utilise toujours ses propres plans pour ce calcul.

Supposons que dep_net(i, action, i) soit l'instance de la classe dep_net que l'agent sujet i a construit pour représenter ses a-dépendances, en utilisant ses propres plans. Du point de vue de l'implémentation, en ce qui concerne son but g, le calcul de la situation de but est très simple :

1. d'abord il vérifie si dans la liste de but-dépendances de dep_net(i, action, i) il en existe une dont le but est g. Si cette but-dépendance n'existe pas, alors la situation de but est NG ;
2. en supposant que une telle instance de la classe goal_dep existe, la deuxième étape consiste à vérifier si la liste de plan-dépendances de cette dernière est vide. Si cette liste est vide, alors la situation de but est NP ;
3. cette liste n'étant pas vide, il faut ensuite vérifier pour chaque instance de la classe plan_dep si sa liste de et-dépendances est vide. S'il existe une telle instance, alors la situation de but est AUT, sinon la situation de but est DEP.

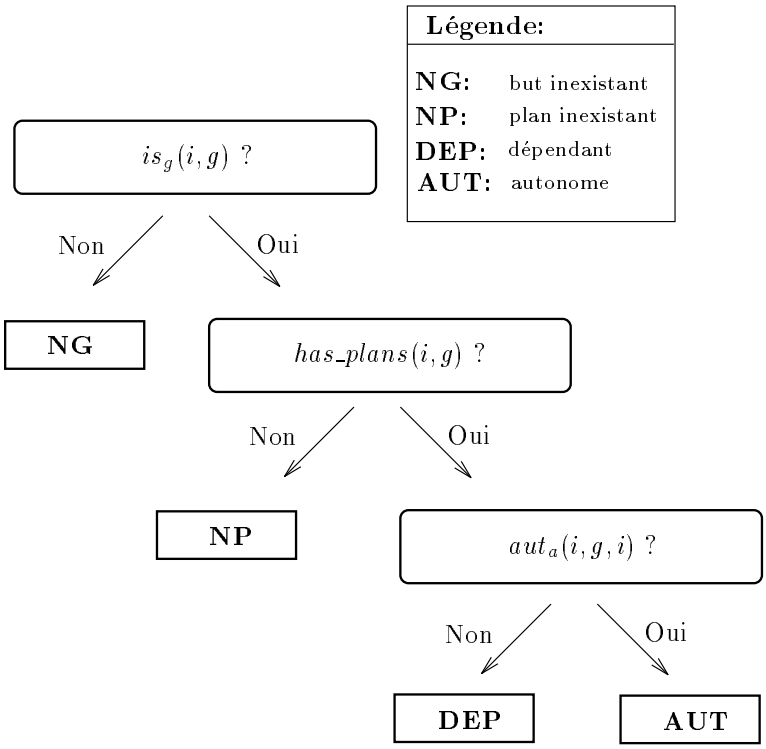


Figure 9.7 : Algorithme pour le calcul des situations de but

9.4 Situations de dépendance

En reprenant de nouveau notre modèle décrit dans le chapitre 5, nous présentons dans la figure 9.8 l'algorithme utilisé par un agent sujet i pour calculer sa situation de dépendance envers l'agent tiers j pour le but g.

Ce calcul est un peu plus compliqué que le précédent, car il doit utiliser quatre réseaux différents :

- dep_net(i, action, i) et dep_net(j, action, i), ces deux premiers pour déduire les dépendances mutuelles et réciproques ;
- dep_net(i, action, j) et dep_net(j, action, j), ces deux derniers pour vérifier si des éventuelles dépendances bilatérales sont localement ou mutuellement reconnues.

Supposons que dep_net(i, action, i) soit l'instance de la classe dep_net que l'agent sujet i a construit pour représenter ses a-dépendances, en utilisant ses propres plans. Du point de vue de l'implémentation, le calcul de sa situation de dépendance envers l'agent tiers j pour le but g est fait de la façon suivante :

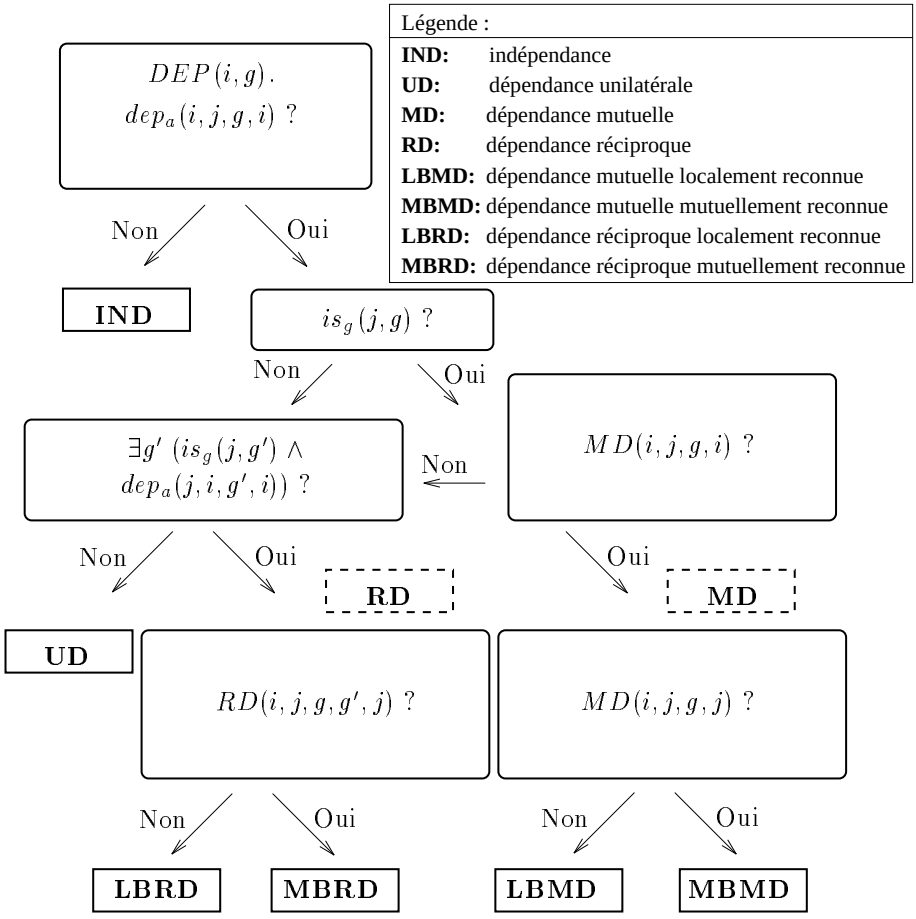


Figure 9.8 : Algorithme pour le calcul des situations de dépendance

1. d'abord il vérifie si il existe un chemin à partir de l'instance de la classe goal_dep relative au but g appartenant à dep_net(i, action, i) qui se termine par une instance de la classe ou_dep relative à j. Si un tel chemin n'existe pas, alors la situation de dépendance est *IND* ;

2. si ce chemin existe, alors il vérifie dans l'instance de la classe ext_descr relative à l'agent tiers j si celui-ci a le but g. S'il n'a pas, il va directement à l'étape 5 ;

3. si l'agent tiers j a le but g, il faut vérifier s'il existe une dépendance mutuelle. Ceci est fait en construisant le réseau dep_net(j, action, i) et en répétant le test décrit dans l'étape 1, cette fois-ci en vérifiant si l'instance de la classe ou_dep est relative à i. S'il n'a pas de dépendance mutuelle, il va directement à l'étape 5 ;

4. dans le cas où une dépendance mutuelle a été détectée dans l'étape précé-

dente, il faut maintenant vérifier s'il s'agit d'une dépendance localement ou mutuellement reconnue. Ceci est fait en construisant les réseaux dep_net(i, action, j) et dep_net(j, action, j) et en répétant les étapes 1, 2 et 3 précédentes. A ce moment là, soit il s'agit d'une *MBMD* soit il s'agit d'une *LBMD* ;

5. si soit l'agent j n'a pas le but g soit il ne dépend pas de i pour le but g, il faut vérifier s'il en dépend pour un autre but. Ceci est fait en vérifiant s'il existe un chemin à partir de dep_net(j, action, i) qui se termine par une instance de la classe ou_dep relative à i. Si un tel chemin n'existe pas, alors la situation de dépendance est *UD* ;

6. dans le cas où une dépendance réciproque a été détectée dans l'étape précédente, il faut maintenant vérifier s'il s'agit d'une dépendance localement ou mutuellement reconnue. Ceci est fait en construisant les réseaux dep_net(i, action, j) et dep_net(j, action, j) et en répétant les étapes 1 et 5. A ce moment là, soit il s'agit d'une *MBRD* soit il s'agit d'une *LBRD*.

Chapitre 10

Simulateur DEPNET

Dans le chapitre 9, nous avons décrit comment nous avons implémenté le mécanisme de raisonnement social. Cependant, nous n'avons pas montré comment ce mécanisme pouvait être exploité.

Comme première application, nous avons développé un *simulateur*, appelé DEPNET. En construisant ce système, notre objectif était de fournir un outil informatique aux chercheurs du domaine de la psychologie sociale pour l'analyse du comportement de micro-sociétés, en particulier en démontrant l'intérêt d'utiliser la théorie de la dépendance pour cette tâche. Par micro-sociétés, nous comprenons un ensemble pas très nombreux d'agents, qui n'ont pas a priori aucune organisation pré-établie entre eux. Avec des résultats fournis par le simulateur, en particulier les réseaux de dépendance et les situations de dépendance, il est possible d'analyser et prédire l'occurrence de plusieurs interactions sociales significatives, telles que nous les avons décrites dans le chapitre 4. Ainsi, comme nous l'avons prévu dans la section 1.4.1, cet outil illustre l'intérêt de notre modèle selon une perspective SS. Nous avons initialement présenté les caractéristiques principales de ce simulateur, ainsi que quelques résultats de simulation, dans [SCDC94] et [CS95].

Ce chapitre est organisé de la façon suivante. Dans la section 10.1, nous décrivons les principales composantes du simulateur. Dans la section 10.2, nous présentons un exemple de simulation d'une fédération de laboratoires de recherche, où nous démontrons que le pouvoir social et les diverses relations de dépendance entre agents changent par l'entrée d'un nouvel agent dans une société. Enfin, dans la section 10.3, nous présentons une discussion à propos de cet outil.

10.1 Description du simulateur

Comme ce simulateur est un outil informatique dont l'objectif est celui de permettre l'analyse de micro-sociétés, on présuppose l'existence d'un observateur extérieur à la société d'agents (l'utilisateur du système) qui a une connaissance complète et correcte des buts, actions, ressources et plans de chacun des agents pour lesquels il veut simuler le comportement social. Pour cette raison, nous

avons fait un choix d'implémentation, c'est-à-dire de ne représenter qu'*une seule description externe* partagée par tous les agents du système. Cela correspond en quelque sorte à considérer les relations de dépendance comme faisant partie d'une réalité *objective*, constituant une base pré-cognitive pour le comportement social, comme nous avons expliqué dans le chapitre 4. Du point de vue du modèle théorique introduit dans le chapitre 5, ce choix n'est pas très compromettant : il ne s'agit que d'adopter l'hypothèse de compatibilité de description externe (équation 8.6) entre tous les agents du système. En outre, comme dans notre modèle un agent prend toujours comme point de départ ses propres plans, nous pouvons simuler avec une seule description externe le mécanisme de raisonnement social de plusieurs agents, simplement en étant attentif à l'ensemble de plans pris au départ. C'est justement cela que nous avons fait dans le cadre de ce simulateur.

Le simulateur, présenté dans la figure 10.1, est composé d'une interface homme-machine capable d'activer les diverses fonctionnalités du mécanisme de raisonnement social. Lorsque ce simulateur a été initialement développé, nous n'avons pas encore introduit dans notre modèle la notion de situation de but. D'ailleurs, c'est justement en l'utilisant que nous nous sommes aperçus qu'une telle notion était nécessaire. Ce fait montre que notre deuxième motivation méthodologique présentée dans la section 1.4.2 est valable : en utilisant un système, nous pouvons détecter qu'il existe des aspects devant être incorporés au modèle à partir duquel le système a été généré. Nous avons fait ensuite le chemin inverse, c'est-à-dire, incorporer ce nouvel aspect dans le simulateur. La même procédure a été adoptée en ce qui concerne le calcul des plans exécutables et de buts réalisables.

Nous avons divisé l'interface du simulateur en six modules principaux :

- *éditeur de la description externe* : ce module permet à l'utilisateur de créer dynamiquement des agents et d'éditer leurs descriptions externes. A tout moment, l'utilisateur peut modifier la composition de ces agents, en leur ajoutant ou retirant des buts, actions, ressources ou plans ;

- *interface pour des réseaux de dépendance* : ce module permet à l'utilisateur de spécifier quel réseau il veut visualiser. Il doit choisir l'agent cible, et aussi spécifier quel est le type de réseau qu'il désire analyser : les réseaux de a-dépendances ou ceux de r-dépendances. En outre, ce module permet à l'utilisateur de visualiser le réseau complet d'un agent (correspondant à tous ses buts) ou bien un sous-réseau particulier ne concernant qu'un seul but de l'agent ;

- *interface pour des situations de dépendance* : ce module permet à l'utilisateur de visualiser les situations de dépendance d'un agent envers d'autres agents pour un de ses buts. Il doit spécifier quel type de situation de dépendance il veut analyser. La sortie du système est différente, complète ou résumée, si la situation calculée par le mécanisme de raisonnement social respectivement correspond ou non à celle désirée par l'utilisateur ;

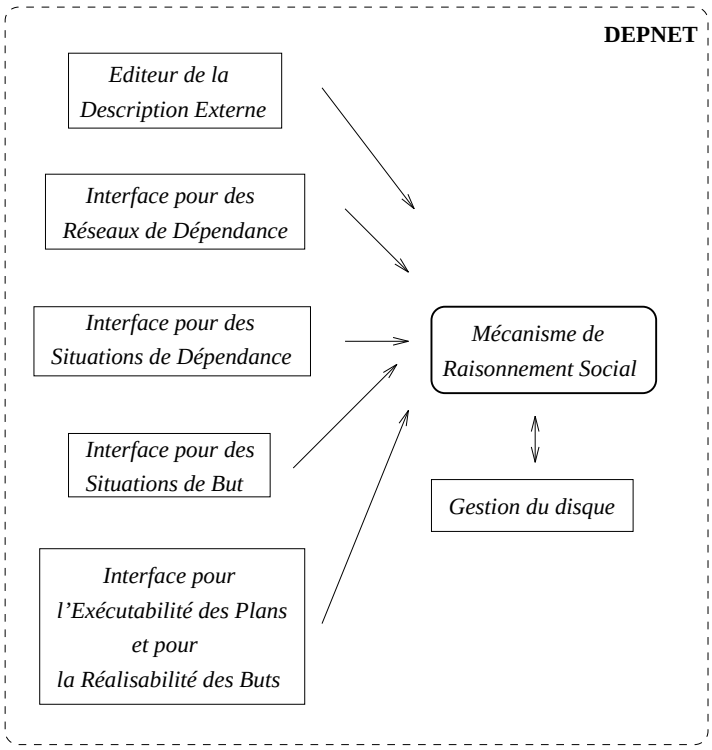


Figure 10.1 : Simulateur DEPNET

- *interface pour des situations de but* : ce module permet à l'utilisateur de visualiser les situations de but d'un agent. Il doit choisir l'agent cible, ainsi que le but qu'il veut analyser. Ce module permet aussi de visualiser les situations de but relatives à tous les buts de l'agent cible ;

- *interface pour l'exécutabilité des plans et pour la réalisabilité des buts* : ce module permet à l'utilisateur de visualiser si les plans d'un agent pour atteindre un certain but sont exécutables ou non et si les buts d'un agent sont réalisables ou non. Comme dans le cas précédent, l'utilisateur peut spécifier un certain but/plan spécifique ou bien signaler au système qu'il veut analyser si tous les buts/tous les plans pour atteindre un certain but sont réalisables/exécutables ;

- *gestion du disque* : ce module permet de lire et écrire le contenu de la description externe d'un disque. De cette façon, l'utilisateur peut simuler plusieurs fois le même exemple de micro-société sans avoir besoin à chaque fois de rentrer le contenu de la description externe dans le système.

Le simulateur DEPNET a été développé sur une station de travail UNIX, en utilisant le langage C++. Son interface est guidée par des menus, dont les fonctions

sont strictement liées aux modules introduits ci-dessus, raison pour laquelle nous n'allons pas la décrire en détail. Cette interface est décrite dans [CS95].

Dans les sections suivantes, nous montrons un exemple d'utilisation de ce simulateur. Nous analysons la dynamique du pouvoir social et des relations de dépendance dans une fédération de laboratoires de recherche, tel que nous l'avons discuté dans le chapitre 6.

10.2 Simulation d'un environnement de recherche

Supposons une fédération de laboratoires de recherche, composée d'un laboratoire d'informatique et d'un autre en électronique. A cause d'une réorganisation interne de la fédération, un chercheur du deuxième laboratoire va migrer vers le premier. Nous allons montrer comment cette migration a un effet très significatif sur les autres agents de ce premier laboratoire, en ce qui concerne leurs réseaux de dépendance et leur pouvoir social dans ce laboratoire[1].

10.2.1 Situation initiale du laboratoire d'informatique

Supposons d'abord une société d'agents correspondant à notre laboratoire d'informatique. Nous avons initialement dans ce laboratoire 4 agents, dont tous ont un certain intérêt pour le domaine des SMA qui'ils ne connaissent pas très bien :

- $ag1$, dont la spécialité est la programmation orientée objet, et qui connaît bien le langage LaTeX. Cet agent a trois buts : écrire un article sur des objets, un deuxième sur la relation entre des objets et des agents et réviser un troisième sur le génie logiciel ;

- $ag2$, dont la spécialité est la simulation sociale, et qui préfère utiliser le logiciel WORD comme traitement de texte. Cet agent a lui aussi trois buts : écrire un article pour la prochaine conférence en simulation sociale, un deuxième sur l'apport des SMA à son domaine et réviser un troisième sur des objets ;

- $ag3$, dont la spécialité sont les systèmes répartis. Cet agent connaît lui aussi le langage LaTeX, et il aussi a trois buts à poursuivre : écrire deux articles, dont un premier sur les systèmes répartis et un deuxième sur l'apport des SMA à son domaine et réviser un troisième sur des objets ;

- $ag4$, dont la spécialité est le génie logiciel. Comme $ag1$ et $ag3$, il utilise le langage LaTeX pour produire ses articles, et lui aussi veut en écrire deux, dont

[1] Une fois de plus, pour une question de simplification, nous n'avons modélisé ce système qu'en utilisant des actions. Par conséquent, nous allons utiliser les termes autonome et dépendant comme des synonymes de a-autonome et a-dépendant.

l'un sur son domaine et l'autre sur la relation entre SMA et génie logiciel. Il a aussi un troisième article à deviser, dans le domaine des systèmes répartis.

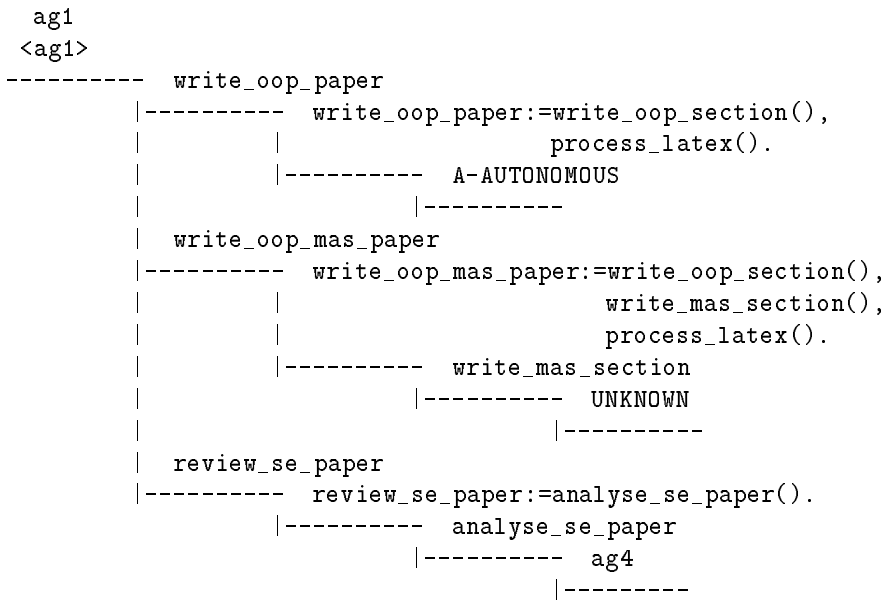
La description externe de ces agents est montré dans l'exemple 10.1, tandis que les divers réseaux de dépendance calculés par le simulateur DEPNET sont détaillés dans l'exemple 10.2 suivant.

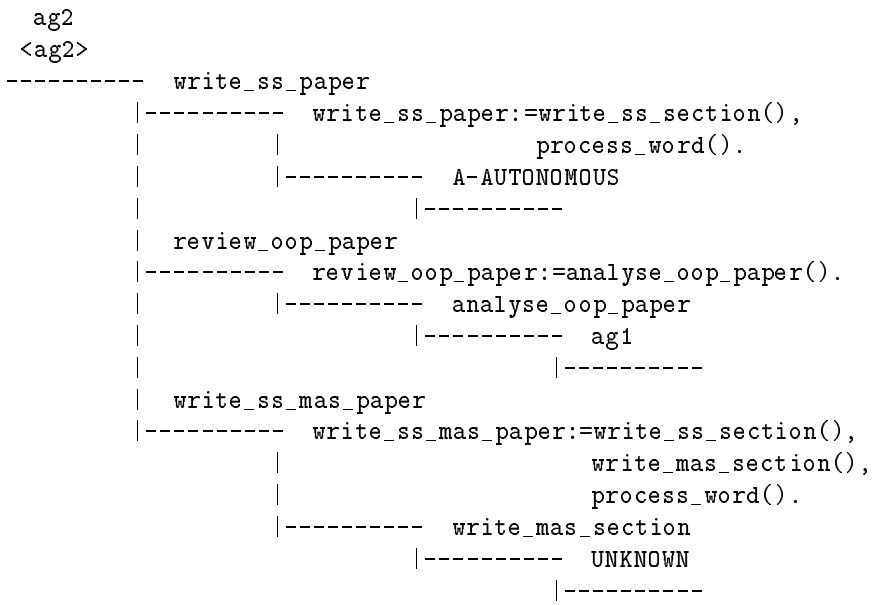
Exemple 10.1 : Description externe initiale du laboratoire d'informatique.

Agent	Buts	Actions	Ressources
$ag1$	write_oop_paper write_oop_mas_paper review_se_paper	write_oop_section analyse_oop_paper process_latex	—
$ag2$	write_ss_paper write_ss_mas_paper review_oop_paper	write_ss_section analyse_ss_paper process_word	—
$ag3$	write_ds_paper write_ds_mas_paper review_oop_paper	write_ds_section analyse_ds_paper process_latex	—
$ag4$	write_se_paper write_se_mas_paper review_ds_paper	write_se_section analyse_se_paper process_latex	—

Agents	Plans		
$ag1$	write_oop_paper()	:=	write_oop_section(), process_latex().
	write_oop_mas_paper()	:=	write_oop_section(), write_mas_section(), process_latex().
	review_se_paper()	:=	analyse_se_paper().
	review_oop_paper()	:=	analyse_oop_paper().
$ag2$	write_ss_paper()	:=	write_ss_section(), process_word().
	write_ss_mas_paper()	:=	write_ss_section(), write_mas_section(), process_word().
	review_oop_paper()	:=	analyse_oop_paper().
	review_ss_paper()	:=	analyse_ss_paper().
$ag3$	write_ds_paper()	:=	write_ds_section(), process_latex().
	write_ds_mas_paper()	:=	write_ds_section(), write_mas_section(), process_latex().
	review_oop_paper()	:=	analyse_oop_paper().
	review_ds_paper()	:=	analyse_ds_paper().
$ag4$	write_se_paper()	:=	write_se_section(), process_latex().
	write_se_mas_paper()	:=	write_se_section(), write_mas_section(), process_latex().
	review_ds_paper()	:=	analyse_ds_paper().
	review_se_paper()	:=	analyse_se_paper().

Exemple 10.2 : Réseaux de dépendance initiaux du laboratoire d'informatique.





```
  ag3
 <ag3>
----------  write_ds_paper
         |----------  write_ds_paper:=write_ds_section(),
         |                          process_latex().
         |----------  A-AUTONOMOUS
                    |----------
         |  review_oop_paper
         |----------  review_oop_paper:=analyse_oop_paper().
         |----------  analyse_oop_paper
                    |----------  ag1
                               |----------
         |  write_ds_mas_paper
         |----------  write_ds_mas_paper:=write_ds_section(),
                    |                       write_mas_section(),
                    |                       process_latex().
                    |----------  write_mas_section
                               |----------  UNKNOWN
                                          |----------

  ag4
 <ag4>
----------  write_se_paper
         |----------  write_se_paper:=write_se_section(),
         |                          process_latex().
         |----------  A-AUTONOMOUS
                    |----------
         |  write_se_mas_paper
         |----------  write_se_mas_paper:=write_se_section(),
         |                          |    write_mas_section(),
         |                          |    process_latex().
         |----------  write_mas_section
                    |----------  UNKNOWN
                               |----------
         |  review_ds_paper
         |----------  review_ds_paper:=analyse_ds_paper().
                    |----------  analyse_ds_paper
                               |----------  ag3
                                          |----------
```

Les situations de but[2], ainsi que les situations de dépendance calculées par le simulateur DEPNET sont présentées dans les exemples 10.3 et 10.4. Dans ce deuxième exemple, nous n'avons représenté que les buts pour lesquels chaque agent est dépendant des autres, représenté par la situation de but *DEP* dans le premier.

[2]Par une question de concision, nous représentons les situations de but et les situations de dépendance sous forme de tableau.

Exemple 10.3 : Situations de but initiales du laboratoire d'informatique.

Agent	But	G-SIT
$ag1$	$write\_oop\_paper$	AUT
	$write\_oop\_mas\_paper$	DEP
	$review\_se\_paper$	DEP
$ag2$	$write\_ss\_paper$	AUT
	$review\_oop\_paper$	DEP
	$write\_ss\_mas\_paper$	DEP
$ag3$	$write\_ds\_paper$	AUT
	$review\_oop\_paper$	DEP
	$write\_ds\_mas\_paper$	DEP
$ag4$	$write\_se\_paper$	AUT
	$write\_se\_mas\_paper$	DEP
	$review\_ds\_paper$	DEP

Exemple 10.4 : Situations de dépendance initiales du laboratoire d'informatique.

D-SIT		Agents			
moi	but	$ag1$	$ag2$	$ag3$	$ag4$
$ag1$	$write\_oop\_mas\_paper$	—	—	—	—
	$review\_se\_paper$	—	—	—	UD
$ag2$	$review\_oop\_paper$	UD	—	IND	—
	$write\_ss\_mas\_paper$	—	—	—	—
$ag3$	$review\_oop\_paper$	UD	IND	—	—
	$write\_ds\_mas\_paper$	—	—	—	—
$ag4$	$write\_se\_mas\_paper$	—	—	—	—
	$review\_ds\_paper$	—	—	UD	—

Nous pouvons faire l'analyse suivante, en prenant compte des propriétés énoncées dans les exemples ci-dessus :

- tous les agents ont le même nombre de buts, et ils sont tous aussi autonomes pour un de leurs buts (celui d'écrire un article sur leur domaine de recherche), comme le montrent les situations de but dans l'exemple 10.3. Même si dans un premier niveau d'analyse cela peut signifier une certaine homogénéité de cette société, cela n'est pas le cas, comme nous le montrons dans la suite ;

- en ce qui concerne le but d'écrire un article dans leur domaine de recherche et les SMA, tous sont dépendants d'une même action $write\_mas\_section$, qui n'est pas disponible dans la société. Cela signifie que *ces buts sont non-réalisables* ;

- un agent ($ag4$) dépend de l'agent $ag3$, un agent ($ag1$) dépend de l'agent $ag4$ et deux agents ($ag2$ et $ag3$) dépendent de l'agent $ag1$. Aucun agent ne dépend de l'agent $ag2$.

  Comme premier essai de quantification des notions de dépendance et de pouvoir social, nous allons adopter le résultat intuitif présenté dans [CC95] : plus le nombre d'agents capables d'entreprendre une certain action a dont l'agent i a besoin est grand, plus la possibilité pour que l'agent i puisse atteindre son but (parce que il existe plusieurs choix de partenaires possibles) est grande et par conséquent plus son degré de dépendance vers chacun des possibles partenaires est petit. De façon analogue, nous pouvons dire que plus le nombre d'agents qui dépendent de l'agent i est grand, plus grand est le pouvoir social de cet agent au sein de cette société. Si nous admettons, de façon préliminaire, que le pouvoir social d'un agent peut être calculé comme étant *le nombre des buts pour lesquels les autres agents dépendent de lui*, nous pouvons constater que le pouvoir social de l'agent $ag1$ est deux fois supérieur à celui des agents $ag3$ et $ag4$, et celui de l'agent $ag2$ est inexistant. Ce fait montre que cette société n'est pas symétrique, par rapport au pouvoir que les agents ont les uns sur les autres. Dans l'exemple, l'agent $ag1$ est celui qui possède le plus de pouvoir social dans cette société ;

- en ce qui concerne les situations de dépendance, nous n'avons pratiquement que les UDs, comme le montre l'exemple 10.4. Un cas intéressant est celui des agents $ag2$ et $ag3$ en ce qui concerne le but $review\_oop\_paper$. En calculant ses situations de dépendance pour ce but, ils détectent les faits suivants : (i) les deux ont le même but, (ii) aucun d'entre eux n'est autonome pour ce but, et (iii) aucun d'entre eux n'est capable d'entreprendre l'action dont l'autre a besoin pour poursuivre ce but. La situation de dépendance calculée par chacun des deux est donc IND[3]. L'existence des UDs peut être expliquée par le fait que nous avons une "boucle" concernant des dépendances : $ag1$ dépend de $ag4$, que pour sa part dépend de $ag3$ que finalement dépend de $ag1$. Si nous considérons que des agents peuvent calculer les dépendances imbriquées (ce que nous ne faisons pas dans ce travail), nous pouvons envisager, que par exemple, $ag1$ puisse envoyer le message suivante à $ag4$: "Si tu m'aides à réviser mon article, j'essaierai d'influencer $ag3$ pour qu'il t'aide, car je peux aussi l'aider à réviser le sien.". Ce type d'exploitation est une extension de la notion de dépendance transitive, présentée dans la section 4.6.3.

[3] A la rigueur, toutes les situations de dépendance représentées par des tirés dans l'exemple 10.4 (ainsi que dans dans les exemples 10.8 et 10.10 introduits dans la suite de cette section) correspondent aussi à la situation IND. Nous ne représentons explicitement que celles où nous avons $IND(i,j,g) \wedge is_g(j,g)$.

10.2.2 Situation initiale du laboratoire d'électronique

Supposons maintenant que notre laboratoire d'électronique, dont les domaines prioritaires sont le traitement du signal et les télécommunications, soit composé par des agents suivants :

- $ag5$, chercheur en SMA, qui veut écrire un article sur la relation entre des SMA et la simulation sociale, une deuxième sur des fondations des SMA, et qui doit aussi réviser une troisième sur des objets, domaine qu'il connaît assez bien ;

- $ag6$, expert dans le domaine des télécommunications, qui a les but suivants : écrire un article sur ce domaine et réviser deux autres, dont l'un sur le génie logiciel et l'autre sur le traitement du signal ;

- $ag7$, chercheur dans le domaine du traitement du signal, qui lui aussi a trois buts : écrire un article sur son domaine et réviser deux autres, dont l'un sur le génie logiciel et l'autre sur les télécommunications.

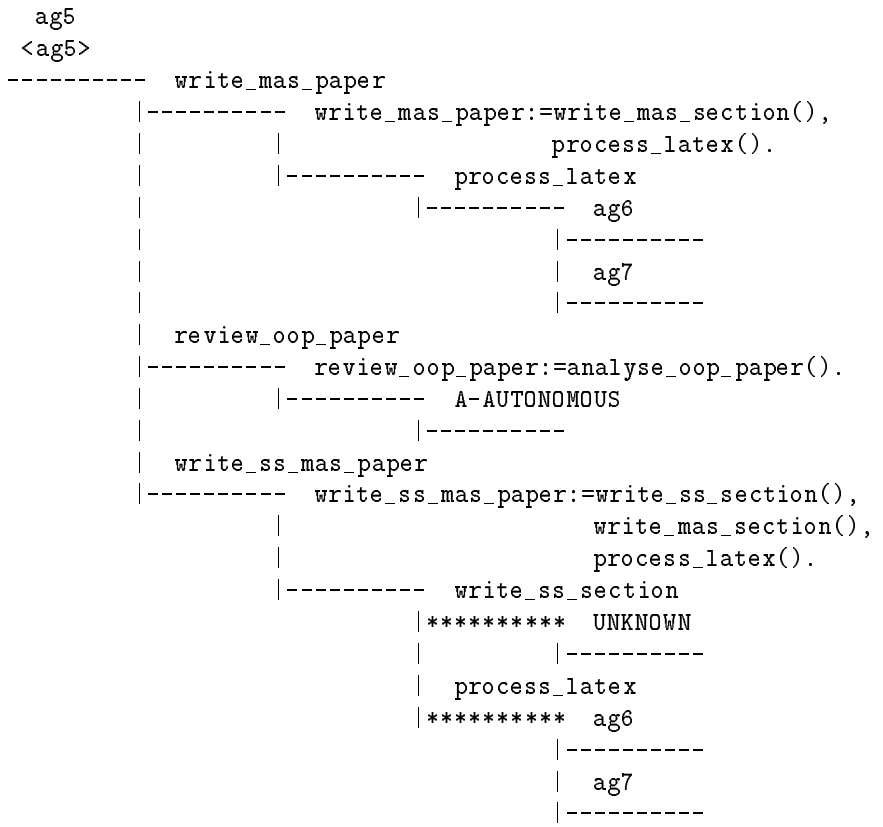
Les agents $ag6$ et $ag7$ connaissent bien le langage LaTeX, tandis que l'agent $ag5$ ne le maîtrise pas. La description externe de cette société est montré dans l'exemple 10.5. Dans les exemples 10.6, 10.7 et 10.8 présentés dans la suite, nous présentons respectivement les réseaux de dépendance, les situations de but et les situations de dépendance de cette société.

Exemple 10.5 : Description externe du laboratoire d'électronique.

Agent	Buts	Actions	Ressources
$ag5$	write_mas_paper write_ss_mas_paper review_oop_paper	write_mas_section analyse_mas_paper analyse_oop_paper	—
$ag6$	write_tel_paper review_sig_paper review_se_paper	write_tel_section analyse_tel_paper process_latex	—
$ag7$	write_sig_paper review_tel_paper review_se_paper	write_sig_section analyse_sig_paper process_latex	—

Agents	Plans		
ag5	write_mas_paper()	:=	write_mas_section(), process_latex().
	write_ss_mas_paper()	:=	write_ss_section(), write_mas_section(), process_latex().
	review_oop_paper()	:=	analyse_oop_paper().
	review_mas_paper()	:=	analyse_mas_paper().
ag6	write_tel_paper()	:=	write_tel_section(), process_latex().
	review_sig_paper()	:=	analyse_sig_paper().
	review_se_paper()	:=	analyse_se_paper().
	review_tel_paper()	:=	analyse_tel_paper().
ag7	write_sig_paper()	:=	write_sig_section(), process_latex().
	review_tel_paper()	:=	analyse_tel_paper().
	review_se_paper()	:=	analyse_se_paper().
	review_sig_paper()	:=	analyse_sig_paper().

Exemple 10.6 : Réseaux de dépendance initiaux du laboratoire d'électronique.



```
 ag6
<ag6>
----------  write_tel_paper
         |----------  write_tel_paper:=write_tel_section(),
         |                            process_latex().
         |----------  A-AUTONOMOUS
                    |----------
         |  review_sig_paper
         |----------  review_sig_paper:=analyse_sig_paper().
         |----------  analyse_sig_paper
                    |----------  ag7
                                 |----------
         |  review_se_paper
         |----------  review_se_paper:=analyse_se_paper().
                    |----------  analyse_se_paper
                                 |----------  UNKNOWN
                                              |----------
```

```
 ag7
<ag7>
----------  write_sig_paper
         |----------  write_sig_paper:=write_sig_section(),
         |                            process_latex().
         |----------  A-AUTONOMOUS
                    |----------
         |  review_tel_paper
         |----------  review_tel_paper:=analyse_tel_paper().
         |----------  analyse_tel_paper
                    |----------  ag6
                                 |----------
         |  review_se_paper
         |----------  review_se_paper:=analyse_se_paper().
                    |----------  analyse_se_paper
                                 |----------  UNKNOWN
                                              |----------
```

Exemple 10.7 : Situations de but initiales du laboratoire d'électronique.

Agent	But	G-SIT
$ag5$	$write\_mas\_paper$	DEP
	$review\_oop\_paper$	AUT
	$write\_ss\_mas\_paper$	DEP
$ag6$	$write\_tel\_paper$	AUT
	$review\_sig\_paper$	DEP
	$review\_se\_paper$	DEP
$ag7$	$write\_sig\_paper$	AUT
	$review\_tel\_paper$	DEP
	$write\_se\_paper$	DEP

Exemple 10.8 : Situations de dépendance initiales du laboratoire d'électronique.

D-SIT		Agents		
moi	but	$ag5$	$ag6$	$ag7$
$ag5$	$write\_mas\_paper$	—	UD	UD
	$write\_ss\_mas\_paper$	—	UD	UD
$ag6$	$review\_sig\_paper$	—	—	$MBRD$
	$review\_se\_paper$	—	—	IND
$ag7$	$review\_tel\_paper$	—	$MBRD$	—
	$review\_se\_paper$	—	IND	—

En considérant les examples présentés ci-dessus, nous pouvons observer que l'agent $ag5$ dépend soit de l'agent $ag6$ soit de l'agent $ag7$ pour atteindre deux de ses buts, à cause de sa méconnaissance du langage LaTeX. En outre, en ce qui concerne son but $write\_ss\_mas\_paper$, il ne peut pas l'atteindre car l'action $write\_ss\_section$ n'est pas disponible. C'est le même cas pour les agents $ag6$ et $ag7$, concernant leur but $review\_se\_paper$. Par contre, ce deux derniers calculent une $MBRD$ entre eux. Finalement, aucun des autres agents ne dépend de l'agent $ag5$, et donc sont pouvoir social est inexistant.

10.2.3 Situation finale du laboratoire d'informatique

Supposons que en conséquence d'une restructuration de la fédération de laboratoires, l'agent $ag5$ ait migré du deuxième vers le premier. Maintenant, le laboratoire d'informatique est composé de 5 agents ($ag1$, $ag2$, $ag3$, $ag4$ et $ag5$), dont nous présentons les nouveaux réseaux de dépendance dans l'exemple 10.9 et les nouvelles situations de dépendance dans l'exemple 10.10. Par concision, nous ne croyons pas qu'il soit nécessaire de présenter une fois de plus la description externe et les situations de but de cette société, car il ne s'agit que d'ajouter à l'exemple 10.1 et à l'exemple 10.3 les données qui concernent l'agent $ag5$ qui ont été initialement présentées respectivement dans les exemples 10.5 et 10.7.

Exemple 10.9 : Réseaux de dépendance finaux du laboratoire d'informatique.

```
 ag1
<ag1>
----------  write_oop_paper
         |----------  write_oop_paper:=write_oop_section(),
         |         |                  process_latex().
         |         |----------  A-AUTONOMOUS
         |                    |----------
         |  write_oop_mas_paper
         |----------  write_oop_mas_paper:=write_oop_section(),
         |         |                       write_mas_section(),
         |         |                       process_latex().
         |         |----------  write_mas_section
         |                    |----------  ag5
         |                               |----------
         |  review_se_paper
         |----------  review_se_paper:=analyse_se_paper().
                    |----------  analyse_se_paper
                               |----------  ag4
                                          |----------
```

```
 ag2
<ag2>
----------  write_ss_paper
         |----------  write_ss_paper:=write_ss_section(),
         |         |                  process_word().
         |         |----------  A-AUTONOMOUS
         |                    |----------
         |  review_oop_paper
         |----------  review_oop_paper:=analyse_oop_paper().
         |         |----------  analyse_oop_paper
         |                    |----------  ag1
         |                               |----------
         |                               |  ag5
         |                               |----------
         |  write_ss_mas_paper
         |----------  write_ss_mas_paper:=write_ss_section(),
                    |                     write_mas_section(),
                    |                     process_word().
                    |----------  write_mas_section
                               |----------  ag5
                                          |----------
```

```
  ag3
 <ag3>
----------  write_ds_paper
         |----------  write_ds_paper:=write_ds_section(),
         |                          process_latex().
         |----------  A-AUTONOMOUS
                    |----------
         |  review_oop_paper
         |----------  review_oop_paper:=analyse_oop_paper().
         |                    |----------  analyse_oop_paper
         |                               |----------  ag1
         |                               |----------
         |                               |  ag5
         |                               |----------
         |  write_ds_mas_paper
         |----------  write_ds_mas_paper:=write_ds_section(),
                    |                       write_mas_section(),
                    |                       process_latex().
                    |----------  write_mas_section
                               |----------  ag5
                                          |----------
```

```
  ag4
 <ag4>
----------  write_se_paper
         |----------  write_se_paper:=write_se_section(),
         |                          process_latex().
         |----------  A-AUTONOMOUS
                    |----------
         |  write_se_mas_paper
         |----------  write_se_mas_paper:=write_se_section(),
         |                    |                  write_mas_section(),
         |                    |                  process_latex().
         |                    |----------  write_mas_section
         |                               |----------  ag5
         |                               |----------
         |  review_ds_paper
         |----------  review_ds_paper:=analyse_ds_paper().
                    |----------  analyse_ds_paper
                               |----------  ag3
                                          |----------
```

```
  ag5
 <ag5>
----------  write_mas_paper
         |----------  write_mas_paper:=write_mas_section(),
         |         |                   process_latex().
         |         |----------  process_latex
         |                   |----------  ag1
         |                             |----------
         |                             |  ag3
         |                             |----------
         |                             |  ag4
         |                             |----------
         |  review_oop_paper
         |----------  review_oop_paper:=analyse_oop_paper().
         |         |----------  A-AUTONOMOUS
         |                   |----------
         |  write_ss_mas_paper
         |----------  write_ss_mas_paper:=write_ss_section(),
                   |                      write_mas_section(),
                   |                      process_latex().
                   |----------  write_ss_section
                             |**********  ag2
                             |         |----------
                             |  process_latex
                             |**********  ag1
                                       |----------
                                       |  ag3
                                       |----------
                                       |  ag4
                                       |----------
```

Exemple 10.10 : Situations de dépendance finales du laboratoire d'informatique.

D-SIT		Agents				
moi	but	*ag1*	*ag2*	*ag3*	*ag4*	*ag5*
ag1	*write_oop_mas_paper*	—	—	—	—	*UD*
	review_se_paper	—	—	—	*UD*	—
ag2	*review_oop_paper*	*UD*	—	*IND*	—	*MBRD*
	write_ss_mas_paper	—	—	—	—	*MBMD*
ag3	*review_oop_paper*	*UD*	*IND*	—	—	*UD*
	write_ds_mas_paper	—	—	—	—	*UD*
ag4	*write_se_mas_paper*	—	—	—	—	*UD*
	review_ds_paper	—	—	*UD*	—	—
ag5	*write_mas_paper*	*UD*	—	*LBRD*	*UD*	—
	write_ss_mas_paper	*UD*	*MBMD*	*LBRD*	*UD*	—

Si nous faisons une comparaison avec la situation initiale décrite auparavant, nous pouvons noter qu'un changement non-négligeable a eu lieu dans cette société :

- en ce qui concerne le but d'écrire un article sur leur domaine de recherche et les SMA, tous les agents qui faisaient partie de cette société dans la situation initiale peuvent maintenant l'atteindre, car maintenant avec l'arrivée de l'agent $ag5$, l'action *write_mas_section* est maintenant disponible dans cette société ;

- l'agent $ag5$ est devenu l'agent dont le pouvoir social est le plus grand dans cette société, car le nombre d'agents qui maintenant dépendent de lui est devenu plus grand que le nombre d'agents qui dépendent de l'agent $ag1$. Son attrait social ("social appeal") a beaucoup changé. Lorsqu'il faisait partie du laboratoire d'électronique, celui-ci n'existait pas, puisque aucun agent ne dépendait de lui ;

- de même, en ce qui concerne l'agent $ag2$, son pouvoir social qui n'existait pas dans la situation initiale, est maintenant non-négligeable, car l'agent $ag5$ dépend d'elle pour atteindre son but *write_ss_mas_paper* ;

- si nous analysons les divers réseaux de dépendance présentés dans l'exemple 10.9, nous pouvons observer de façon intuitive que dans cette nouvelle situation, les agents $ag3$ et $ag2$ sont moins dépendants de l'agent $ag1$ que dans la situation initiale. Maintenant, ils ont *un choix* en ce qui concerne l'action *analyse_oop_paper*, car l'agent $ag5$ peut l'entreprendre aussi. Ce fait explique la raison pour laquelle le pouvoir social de l'agent $ag1$ est maintenant plus petit que celui correspondant à la situation initiale ;

- en ce qui concerne les situations de dépendance, elles sont maintenant très différentes de celles présentées dans l'exemple 10.4 concernant la situation initiale, quand il n'y avait pratiquement que des *UD*s. Par exemple, les agents $ag5$ et $ag2$ calculent une *MBMD* l'un envers l'autre pour le but *write_ss_mas_paper*, car les deux savent qu'ils dépendent l'un de l'autre pour atteindre ce but. Les situations de dépendance calculées par les agents $ag5$ et $ag3$ l'un envers l'autre ne sont pas symétriques : comme l'agent $ag5$ a un plan pour le but *review_oop_paper*, il sait que l'agent $ag3$ dépend de lui pour ce but, à cause de l'action *analyse_oop_paper*, pouvant inférer une *LBRD*. Par contre, comme l'agent $ag3$ n'a aucun plan pour atteindre les buts de l'agent $ag5$, il ne peut inférer qu'une *UD* a entre eux. C'est aussi pour cette raison que l'agent $ag5$ calcule une *LBRD* et non une *MBRD*[4].

[4]Il faut rappeler que dans ce système, dont nous avons discuté dans la section 10.1, nous avons adopté l'hypothèse de compatibilité de description externe entre tous les agents qui en font partie.

10.2.4 Situation finale du laboratoire d'électronique

La situation finale du laboratoire d'électronique ne change pas vis-à-vis de sa situation initiale, car le départ de l'agent $ag5$ n'a pas aucun impact sur les agents $ag6$ et $ag7$. Son pouvoir social n'existait pas, puisque ces deux derniers ne dépendaient de lui pour aucun de leurs buts. Par conséquent, la nouvelle description externe, les nouveaux réseaux de dépendance et les nouvelles situations de dépendance de cette société sont identiques à ceux présentés respectivement dans les exemples 10.5, 10.6 et 10.8, en supprimant les informations concernant l'agent $ag5$.

10.3 Discussion

Jusqu'à présent, nous ne connaissons pas d'autres simulateurs existants fondés sur la notion de dépendance sociale. Pour cette raison, il nous est impossible de comparer sa performance avec celle obtenue par d'autres outils similaires.

Néanmoins, en ce qui concerne les fonctionnalités de cet outil, nous croyons qu'ils existent beaucoup d'améliorations envisageables :

- l'interface du système peut être plus *conviviale*. En effet, comme les utilisateurs potentiels du système n'avaient pas un accès facile à des stations de travail lors de sa conception, nous avons conçu une interface basée sur des menus, mais sans utiliser des ressources graphiques très sophistiquées ;

- pour mieux évaluer les relations de dépendance et de pouvoir, il serait intéressant de pouvoir les *quantifier*. En effet, dans la version actuelle du simulateur, il existe déjà un calcul qui est fait dans ce sens, mais à notre avis il est un peu naïf, puisqu'il ne considère pas l'exécutabilité des plans, telle que nous l'avons introduit dans la section 6.3. Nous croyons que cet aspect doit mériter beaucoup de réflexion de notre part dans l'avenir ;

- le *calcul du pouvoir social*, tel que nous avons fait dans l'exemple de la fédération de laboratoires de recherche, n'est pas fait de façon automatique dans la version actuelle du système. Cet aspect peut être justifié en partie par l'absence d'une quantification de ces relations, comme nous venons de le décrire. Sans doute, une amélioration du système dans ce sens est hautement désirable.

Les points abordés ci-dessus font partie de nos perspectives, présentées dans la section 12.2.

Chapitre 11

Système DEPINT

Dans ce chapitre, nous décrivons une deuxième application, appelée DEPINT, que nous avons développée pour illustrer l'intérêt de notre mécanisme de raisonnement social, cette fois-ci selon une perspective RP, comme nous l'avons introduit dans la section 1.4.1.

L'objectif du système DEPINT est de mettre en œuvre les aspects théoriques que nous avons développés dans les chapitres 6, 7 et 8, c'est-à-dire, montrer comment des agents peuvent concevoir dynamiquement une organisation dans un contexte de SMA ouverts, comment ils peuvent s'adapter à la dynamique d'une société ouverte, et comment ils peuvent réviser les informations incomplètes ou incorrectes qu'ils possèdent les uns sur les autres.

Ce chapitre est organisé de la façon suivante. Dans la section 11.1, nous décrivons la philosophie et les composantes de la plate-forme multi-agents du groupe MAGMA, de laquelle nous avons utilisé le système de gestion de protocoles d'interaction pour implémenter les échanges de messages entre les agents du système DEPINT. Ensuite, dans la section 11.2, nous détaillons les protocoles conçus et/ou utilisés pour accomplir nos objectifs. L'implémentation du modèle d'agent est présenté dans la section 11.3, où nous explicitons les simplifications adoptées par rapport au modèle présenté dans la section 3.4. Quelques exemples d'utilisation du système sont présentés dans la section 11.4, où nous illustrons comment les divers comportements spécifiés dans les sections 6.2.2 (choix de buts), 6.3.2 (choix de plans), 7.1.4 (choix de partenaires), 7.2.2 (acceptation de partenaire) et 8.5.2 (choix de contexte) sont implémentés au sein des agents de ce système. Enfin, dans la section 11.5, nous présentons une discussion à propos de ce système.

11.1 Plate-forme du groupe MAGMA

Dans les sous-sections suivantes, nous présentons les caractéristiques principales de la plate-forme multi-agents du groupe MAGMA. Nous nous concentrons plutôt sur la partie concernant les interactions entre agents, puisque nous l'avons

utilisée pour structurer les échanges d'informations entre les agents du système DEPINT. Nous comparons aussi l'approche du groupe MAGMA avec celle connue comme "agent-oriented programming" [Sho91].

11.1.1 Description générale

Dans la figure 11.1, nous présentons l'environnement de conception de SMA en cours de développement au sein du groupe MAGMA [Dem95]. L'objectif de cet environnement est de permettre à un concepteur de SMA de réutiliser des prototypes d'agents, d'organisations et d'interactions existants pour développer des applications de façon plus efficace, comme nous l'avons discuté dans la section 2.2.1. L'idée fondamentale est que la conception des agents, des organisations, et des protocoles d'interaction consiste en des activités disjointes. Actuellement, le modèle d'agent ASIC [Boi93, BD94b] et le système de gestion de protocoles d'interaction décrit dans [PBS93] sont déjà intégrés au sein de cette plate-forme. Nous croyons que notre mécanisme de raisonnement social s'insère parfaitement dans le cadre de cette plate-forme, puisque son exploitation permet aux agents de former dynamiquement des coalitions[1]. Nous n'allons pas nous étendre sur cet environnement, une description plus détaillée peut être trouvée dans [Dem95].

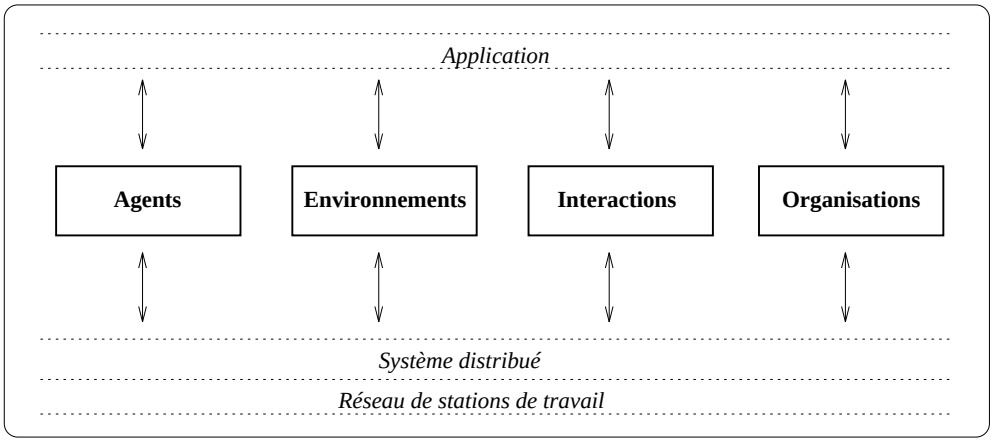


Figure 11.1 : Plate-forme du groupe MAGMA

11.1.2 Interactions entre agents

Nous utilisons dans le système DEPINT le langage d'interaction entre agents proposé dans [Dem95]. Ce langage définit le vocabulaire et la syntaxe communs à tous les agents du système. La syntaxe de ce langage est la suivante :

$$\langle interaction \rangle \quad ::= \quad \langle communication \rangle \, \langle multi - agents \rangle \, \langle application \rangle$$

[1] Rappelons nous que, selon la définition présentée dans la section 3.2, une coalition n'est qu'une organisation conçue dynamiquement.

où

- le champ $\langle communication \rangle$ est lié aux aspects d'implémentation de la couche distribuée sous-jacente. Il est composé de l'émetteur du message, du récepteur du message, d'une identification du message, d'une identification du médium de communication utilisé[2] et du mode de communication (synchrone ou asynchrone) :

$$\langle communication \rangle \quad ::= \quad \langle from \rangle \, \langle to \rangle \, \langle id \rangle \, \langle via \rangle \, \langle mode \rangle$$

- le champ $\langle multi-agents \rangle$ est lié à la dimension multi-agents. Afin d'utiliser des langages des domaines d'application simples, le groupe MAGMA a développé un langage multi-agents assez complexe, composé par le type, la nature et la force du message, ainsi que par une identification du protocole d'interaction utilisé et la position du destinataire dans le protocole lorsqu'il reçoit le message[3] [Dem95]. Pour le type d'interaction, on a utilisé les primitives proposées dans [Gas91] : *request*, *answer* et *inform*. En ce qui concerne la force, on a adopté un sous-ensemble des tons de communication proposés dans [CD90], allant de *commanding* (priorité maximal) à *informing*, qui caractérise une simple échange d'information. Enfin, pour la nature [Boi93], on spécifie quel est le statut de l'information envoyée : *dec* (buts à atteindre), *ada* (plans à exécuter), *com* (actions à entreprendre) et *obs* (échange d'hypothèses).

$$\langle multi-agents \rangle \quad ::= \quad \langle type \rangle \, \langle strength \rangle \, \langle nature \rangle \, \langle protocol \rangle \, \langle position \rangle$$

- le champ $\langle application \rangle$ doit être instancié pour l'application en question. Il comporte les termes propres au domaine. Comme nous le verrons dans la section suivante, dans le contexte du groupe MAGMA, les interactions sont structurées dans des protocoles, et ceux-ci ne sont pas opérationnels sans la définition d'un langage qui exprime la sémantique du domaine en question.

11.1.3 Protocoles d'interaction

Une fois défini le langage d'interaction, il est nécessaire d'établir le déroulement de ces interactions pendant l'exécution d'un système. Nous avons adopté dans le système DEPINT le modèle proposé dans [PBS93, DBK94a, DBK94b, Dem95] pour représenter les *protocoles d'interaction*. L'idée principale des protocoles, inspirés par les travaux sur les systèmes gouvernés par lois ("law-governed systems") [Min89, MR89], est de contraindre les possibles interactions entre agents. Un agent

[2] Dans certains systèmes, plusieurs types de médium peuvent être utilisés, par exemple l'un à très haut débit pour les importants volumes de données, et un autre pour les simples échanges de messages par mécanisme de boîte à lettres [KDEQ95].

[3] La notion de protocole d'interaction est présentée dans la sous-section suivante.

ne peut pas envoyer n'importe quelle information à n'importe quel agent n'importe quand : nous imposons une structuration dans les échanges d'informations.

Un protocole est un graphe d'état dans lequel un agent évolue en fonction des types de messages reçus [KDEQ95]. Il définit un ensemble d'états possibles (représentés par des cercles), ainsi qu'un ensemble de transitions possibles (représentées par des arcs). Les agents changent d'état au fur et à mesure qu'ils interagissent.

Dans [PBS93], nous présentons l'implémentation d'un système de gestion de protocoles, qui permet aux agents de raisonner explicitement sur le déroulement des interactions et de choisir éventuellement une option, parmi plusieurs, afin de continuer une conversation courante. Nous avons utilisé ce système de gestion de protocoles dans le système DEPINT.

11.1.4 Intégration orientée agent

Pour conclure cette section, nous ne pouvons pas nous empêcher de comparer l'approche du groupe MAGMA avec celle connue comme *programmation orientée agent* ("agent-oriented programming") [Sho91, Tho93]. Dans ces autres travaux, la notion d'agent est considérée comme une extension de la notion d'objet, plus particulièrement selon un point de vue structurel. Cette extension est caractérisée en spécifiant la sémantique qui doit être associée aux données (des croyances, des intentions, des plans) et en introduisant un typage dans les primitives de communication tel que nous l'avons discuté dans la section 2.2.1. Cette approche revient à caractériser les travaux en SMA comme faisant partie de la programmation détaillée ("programming-in-the-small"). Nous reprenons les critiques formulées dans [Dem95] et nous considérons que cette approche présente quelques inconvénients, tels que :

- l'existence d'un modèle abstrait unique d'agent, or nous considérons qu'au niveau de la société, nous pouvons bien avoir des agents hétérogènes, avec des mécanismes internes différents ;
- ce modèle abstrait est strictement lié à un modèle d'implémentation unique, ce qui ne rend pas possible d'implémenter des agents en utilisant des langages de programmation différents.

Nous croyons que les travaux en SMA concernent plutôt la programmation globale ("programming-in-the-large") [Tic92]. Pour cette raison, nous considérons plus adéquat d'utiliser le terme *intégration orientée agent* à la place du terme programmation orientée agent pour caractériser le cadre de travail du groupe MAGMA.

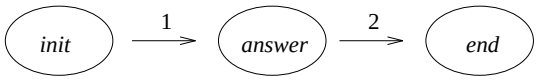
D'autres exemples de travaux qui partagent comme nous une vision d'intégration orientée agent peuvent être trouvés dans [OZT90, Ber93, Arm93, OAA+94], en qui concerne les aspects génie logiciel et dans la littérature sur le projet ARCHON [JW92].

11.2 Protocoles d'interaction du système DEPINT

Dans les sous-sections suivantes, nous présentons les protocoles d'interaction que nous utilisons dans le système DEPINT. Par souci de clarté, nous ne représentons que le destinataire du message[4] du champ $\langle communication \rangle$.

11.2.1 Protocole de présentation

Dans la figure 11.2, nous présentons le protocole de présentation d'un agent. Lorsqu'un agent entre dans la société, il envoie à tous les autres un message signalant son arrivée, avec le contenu de son entrée de description externe. En recevant un tel message, un agent lui répond, en lui envoyant sa propre entrée de description externe.



1. (all) (inform, informing, obs, depintpresentation, answer) (introduction, data)
2. (you) (inform, informing, obs, depintpresentation, end) (presentation, data)

Figure 11.2 : Protocole de présentation

Pour un message d'introduction ou de présentation, le champ $\langle data \rangle$, qui correspond au contenu de la description externe[5], est le suivant :

$$\langle data \rangle \quad ::= \quad \langle agent\_id \rangle \, \langle goals \rangle \, \langle actions \rangle \, \langle plans \rangle \, \langle resources \rangle$$

où :

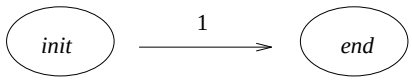
$$\begin{array}{lll}
\langle agent_id \rangle & ::= & \langle agent_name \rangle \, \langle host_name \rangle \, \langle agent_pid \rangle \\
\langle goals \rangle & ::= & \langle n_g \rangle \, \langle id(g_1) \rangle \, \langle w(g_1) \rangle \, \ldots \langle id(g_n) \rangle \, \langle w(g_n) \rangle \\
\langle actions \rangle & ::= & \langle n_a \rangle \, \langle id(a_1) \rangle \, \langle c(a_1) \rangle \, \ldots \langle id(a_n) \rangle \, \langle c(a_n) \rangle \\
\langle plans \rangle & ::= & \langle n_p \rangle \langle id_g(p_1) \rangle \, \langle n_a(p_1) \rangle \, \langle id_a(i_1) \rangle \, \langle n_r(i_1) \rangle \, \langle id(r_1^\star) \rangle \, \ldots \langle id(r_n^\star) \rangle \\
 & & \quad \ldots \qquad \langle id_a(i_n) \rangle \, \langle n_r(i_n) \rangle \, \langle id(r_1^\star) \rangle \, \ldots \langle id(r_n^\star) \rangle \\
 & & \quad \ldots \\
 & & \langle id_g(p_n) \rangle \, \langle n_a(p_n) \rangle \, \langle id_a(i_1) \rangle \, \langle n_r(i_1) \rangle \, \langle id(r_1^\star) \rangle \, \ldots \langle id(r_n^\star) \rangle \\
 & & \quad \ldots \qquad \langle id_a(i_n) \rangle \, \langle n_r(i_n) \rangle \, \langle id(r_1^\star) \rangle \, \ldots \langle id(r_n^\star) \rangle \\
\langle resources \rangle & ::= & \langle n_r \rangle \, \langle id(r_1) \rangle \, \langle c(r_1) \rangle \, \ldots \langle id(r_n) \rangle \, \langle c(r_n) \rangle
\end{array}$$

[4]Le terme *all* désigne une diffusion de message à tous les agents, tandis que le terme *you* désigne l'agent émetteur auquel l'agent doit envoyer une réponse.

[5]Par concision, nous ne répétons pas ici la signification des termes de ce langage, puisque celle-ci est décrite en détail dans la section 5.1.1.

11.2.2 Protocole de sortie

Dans la figure 11.3, nous présentons le protocole de sortie. Lorsqu'un agent sort de la société, il envoie à tous les autres un message signalant son départ. En recevant un tel message, un agent retire les informations relatives à ce dernier de sa description externe.

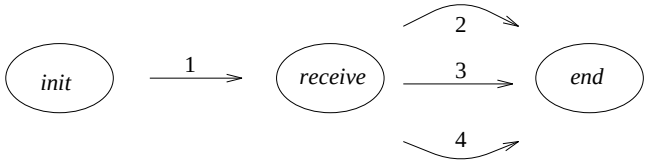


1. (all) (inform, informing, obs, depintexit, end) (exit)

Figure 11.3 : Protocole de sortie

11.2.3 Protocole de formation de coalitions

Dans la figure 11.4, nous présentons le protocole de formation de coalitions.



1. (you) (request, dec, bargaining, depintproposition, receive) (proposition, data)
2. (if error) (you) (answer, warning, obs, depintproposition, end) (revision, data)
3. (if (!error and !best_option)) (you) (answer, informing, dec, depintproposition, end) (refusal)
4. (if(!error and best_option)) (you) (answer, informing, dec, depintproposition, end) (acceptance)

Figure 11.4 : Protocole de formation de coalitions

Nous pouvons remarquer que dans ce protocole il existe trois interactions possibles entre les états *receive* et *end*. Le choix entre ces interactions dépend de deux variables, *error* et *best_option*, qui doivent être instanciées par les agents dynamiquement, lorsqu'ils reçoivent des propositions de coalition.

La variable *error* a la valeur vrai s'il existe une inconsistance au niveau de la société, dans le sens où nous l'avons introduit dans le chapitre 8. La variable *best_option* a la valeur vrai si le récepteur croit que le proposant appartient à l'ensemble de partenaires dont la situation de dépendance est la plus favorable pour lui, en conformité avec le critère d'acceptation présenté dans la section 7.2.

Proposition d'une coalition

Pour un message de proposition, la syntaxe du champ $\langle data \rangle$ est la suivante :

$$\langle data \rangle \quad ::= \quad \langle dsit \rangle \, \langle my\_goal \rangle \, \langle needed\_action \rangle \langle offered\_action \rangle \, \langle your\_goal \rangle$$

où

$$\langle dsit \rangle \quad ::= \quad UD \mid LBMD \mid MBMD \mid LBRD \mid MBRD$$

Le champ $\langle dsit \rangle$ contient la situation de dépendance inférée par le proposant. Les autres champs contiennent respectivement le but que le proposant veut atteindre, l'action requise, le but et l'action offerts. Les champs $\langle offered\_goal \rangle$ et $\langle offered\_action \rangle$ ne sont pas remplis dans le cas d'une dépendance unilatérale (UD).

Révision de croyances

Ayant détecté une inconsistance au niveau de la société, un agent envoie au proposant un message signalant que ce dernier a une représentation incomplète ou incorrecte de lui. Pour un message de révision, le champ $\langle data \rangle$ est défini de la façon suivante :

$$\langle data \rangle \quad ::= \quad \langle entry\_type \rangle \, \langle revision\_type \rangle \, \langle element \rangle$$

où :

$$\begin{aligned} \langle entry\_type \rangle \quad &::= \quad E\_goal \mid E\_action \mid E\_resource \mid E\_plan \\ \langle revision\_type \rangle \quad &::= \quad Incomplete \mid Incorrect \end{aligned}$$

Le champ $\langle element \rangle$ contient l'information particulière, qui doit être révisée.

Acceptation d'une coalition

Si aucune inconsistance au niveau de la société existe et si la variable *best_option* est vrai, alors l'agent accepte de participer à la coalition. Le terme *acceptation* désigne cette situation.

Evidemment, une fois une coalition établie, il faut envisager un autre protocole. Par exemple, pour l'action de la coalition telle que nous l'avons décrite dans la section 3.2, ce nouveau protocole servirait à coordonner l'action conjointe des agents faisant partie de la coalition. Néanmoins, puisque notre intérêt se limite à la formation de coalitions, nous ne présentons pas dans cette thèse un tel protocole.

Réfutation d'une coalition

Si aucune inconsistance au niveau de la société existe, mais que la variable *best_option* est fausse, alors le destinataire refuse de faire partie de la coalition. Comme dans le cas précédent, cette situation est signalée au proposant par le terme $refusal$. Lorsque le proposant reçoit un tel message, il essaie de contacter d'autres partenaires pour atteindre son but.

11.3 Implémentation du modèle d'agent

Dans les sous-sections suivantes, nous soulignons quelques aspects essentiels concernant l'implémentation du modèle d'agent introduit dans la section 3.4. En particulier, nous énumérons quelques simplifications qui ont étés adoptées, et nous présentons les cycles d'activation des mécanismes internes.

11.3.1 Quelques simplifications

Dans le cadre du système DEPINT, nous avons adopté les simplifications suivantes:

- les agents n'ont pas de mécanisme de perception, c'est l'utilisateur du système qui simule cette source d'information ;
- de même, l'évolution des connaissances de l'agent, relatives au domaine d'application, sont simulées à l'aide d'une interface avec l'utilisateur, en lui laissant le soin d'inférer de nouvelles connaissances et de les communiquer au système ;
- nous nous limitons aux cas des coalitions entre deux partenaires[6]. Dans la version actuelle du système, les dépendances multi-partites introduites dans la section 4.6.1 ne sont pas traitées ;
- nous ne traitons pas la concurrence de propositions de coalition. Ainsi, nous avons défini deux types possibles de comportement pour un agent: *actif* et *passif*. Dans le premier cas, l'agent a l'initiative de proposer une coalition aux autres, en utilisant son mécanisme de raisonnement social pour choisir les buts, plans et partenaires. Pour cela, il utilise les critères définis dans les sections 6.2.2 (choix de buts), 6.3.2 (choix de plans) et 7.1.4 (choix de partenaires). Dans le deuxième cas, l'agent active les mécanismes de perception et d'inférence sur le domaine, et il se limite à répondre aux propositions de coalition envoyées par des autres, en utilisant le critère d'acceptation de partenaires défini dans la section 7.2.2. La procédure de révision de croyances est activée dans les deux types de comportements énumérés ci-dessus. C'est l'utilisateur du système qui choisit le comportement qu'un agent doit avoir à un instant donné ;
- le critère de choix de contexte n'est activé que dans le cas d'une contraction de la base de croyances. Autrement dit, nous avons utilisé implicitement une négation par échec en ce qui concerne la description externe.

[6]Cette limitation est due seulement à des restrictions de temps. L'extension du cas de deux partenaires au cas de plusieurs partenaires est évidente: il suffit que l'agent attend la réponse de tous, et dans le cas où un de ces partenaires refuse de participer à la coalition, l'agent doit essayer d'envoyer une proposition à un autre. Si des partenaires possibles n'existent pas, alors l'agent doit essayer un autre plan, et notifier ceux qui ont accepté du fait que la coalition n'est plus valable.

Dans la version actuelle du système, nous avons aussi simplifié l'implémentation du protocole de présentation: le champ $\langle data \rangle$ n'est pas envoyé, et les agents lisent les entrées de description externe les uns des autres dans des fichiers dont les noms sont les noms des agents concernés. De cette façon, dans la version courante, deux agents ne peuvent pas avoir le même nom.

11.3.2 Cycles d'activation des mécanismes internes

Comme nous venons de l'expliquer dans la section précédente, les agents du système DEPINT peuvent avoir un comportement actif ou passif. Pour chacune de ces situations, nous avons défini un cycle d'activation des mécanismes internes.

Comportement Actif

La figure 11.5 présente le cycle d'activation des mécanismes internes dans le cas où l'agent a un comportement actif.

Par concision, nous n'allons pas répéter ici la branche descendante d'activation des mécanismes internes, puisque elle correspond entièrement à la gestion de choix de buts, plans et partenaires, décrites respectivement dans les sections 6.2.3, 6.3.3 et 7.1.5. .

Une fois envoyée une proposition de coalition, un agent peut recevoir trois réponses différentes, conformément à ce que nous avons montré dans la section 11.2.3 :

1. si l'agent reçoit un message de révision, alors les mécanismes de raisonnement, décision et révision sont activés afin d'éliminer l'inconsistance détectée. Une fois cette procédure accomplie, l'agent recommence à raisonner sur les partenaires, en enlevant de la liste de partenaires possibles l'émetteur du message de révision, puisque celui-ci ne peut pas entreprendre l'action requise[7] ;

2. si l'agent reçoit un message de refus, alors il recommence à raisonner sur les partenaires, en enlevant aussi de la liste de partenaires possibles l'émetteur du message de refus ;

3. si l'agent reçoit un message d'acceptation, alors la coalition est formée et il retourne à l'état initial du système.

Dans le cas où le message reçu est du type révision ou refus, la liste de partenaires possibles peut devenir vide. Lorsque cette situation se produit, l'agent recommence à raisonner sur les plans, en essayant de trouver un autre plan exécutable pour le but poursuivi. Si la liste de plans possibles devient vide aussi, alors l'agent recommence à raisonner sur les buts, en essayant de trouver un autre but réalisable. Finalement, si plus aucun but n'est réalisable, alors l'agent retourne à l'état initial du système.

[7] Ce fait est justifié par les principes de la sincérité (P2) et de l'auto-connaissance (P3).

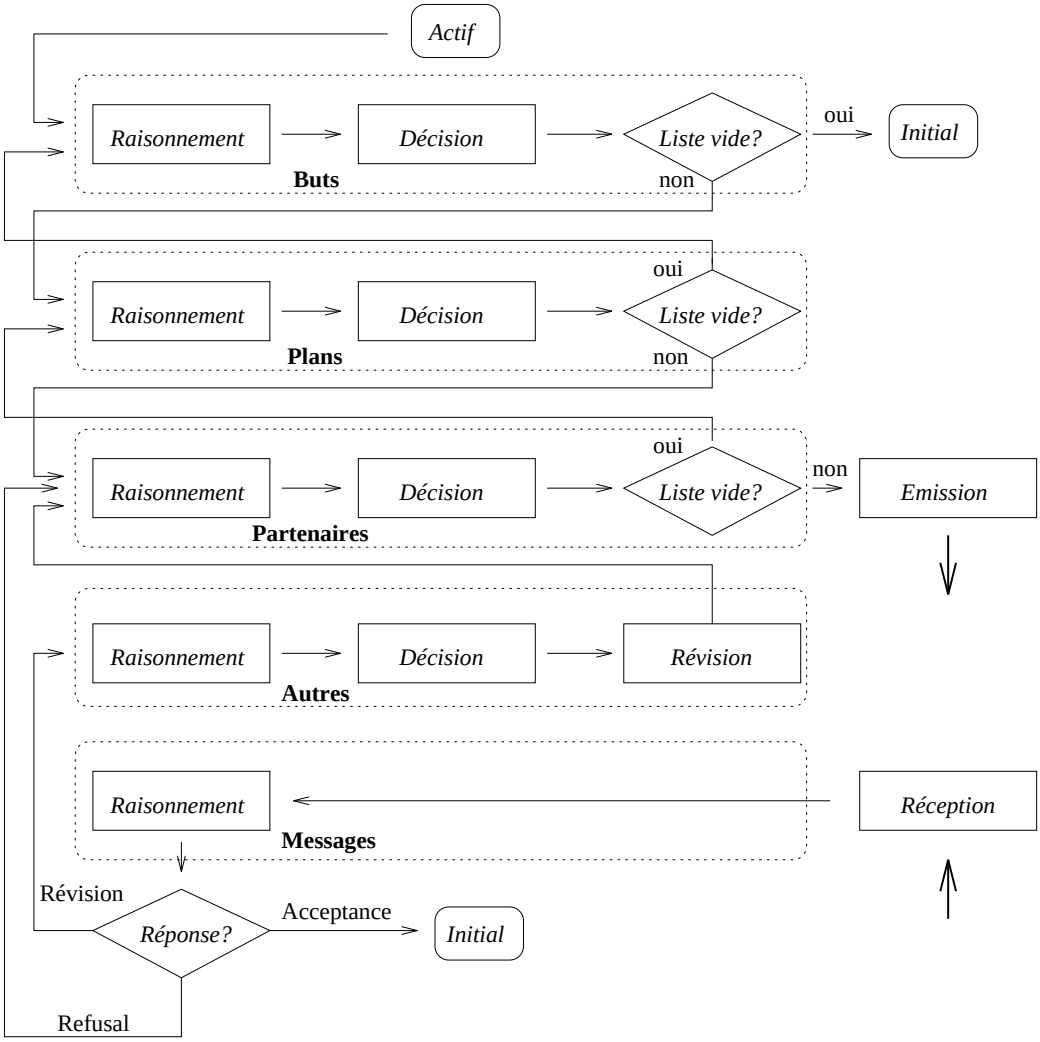


Figure 11.5 : Comportement actif

Comportement Passif

Nous présentons dans la figure 11.6 le cycle d'activation des mécanismes internes lorsque l'agent a un comportement passif.

Dans cette situation, nous avons les étapes suivantes:

1. tout d'abord, le système demande à l'utilisateur s'il veut activer le mécanisme d'inférence sur le domaine[8]. Si la réponse est oui, l'utilisateur entre les données et le système active une procédure de révision de croyances ;

2. la même procédure est utilisée pour simuler la perception. La seule différence entre ces deux étapes réside dans le fait que la source d'information attribuée à une donnée insérée est différente ;

3. la prochaine étape consiste à vérifier si des messages existent pour l'agent. Dans cette situation, quatre types de message peuvent arriver :

   (a) un message de présentation, correspondant à la première transition du protocole de présentation (section 11.2.1). Dans ce cas, l'agent met à jour sa description externe, en y ajoutant les informations concernant le nouvel agent et lui envoie une réponse (deuxième transition du protocole) ;

   (b) une réponse au message de présentation que l'agent lui-même a envoyé lorsqu'il est entré dans le système. Dans cette situation, l'agent met à jour sa description externe, comme dans le cas précédent, mais il n'envoie aucun message ;

   (c) un message de sortie, correspondant au protocole de sortie (section 11.2.2). L'agent doit alors retirer de sa description externe les informations concernant l'agent sortant ;

   (d) une proposition de coalition (section 11.2.3). Dans cette situation, l'agent raisonne et décide sur le message, en utilisant le critère d'acceptation de partenaires présenté dans la section 7.2. Eventuellement, si la situation de but de l'agent est *DEP*, les étapes de raisonnement et de choix sur les plans et les partenaires sont activées, comme dans le cas du comportement actif, afin de comparer la situation de dépendance offerte par le proposant avec celle que l'agent lui-même choisirait s'il devait poursuivre ce but ;

4. la réception des messages est répétée jusqu'à ce qu'il n'en reste plus.

[8]Comme nous avons déjà expliqué auparavant, dans le cadre du système DEPINT ce mécanisme n'est qu'une interface pour l'édition de la description externe. D'ailleurs, la même remarque est valable en ce qui concerne la perception.

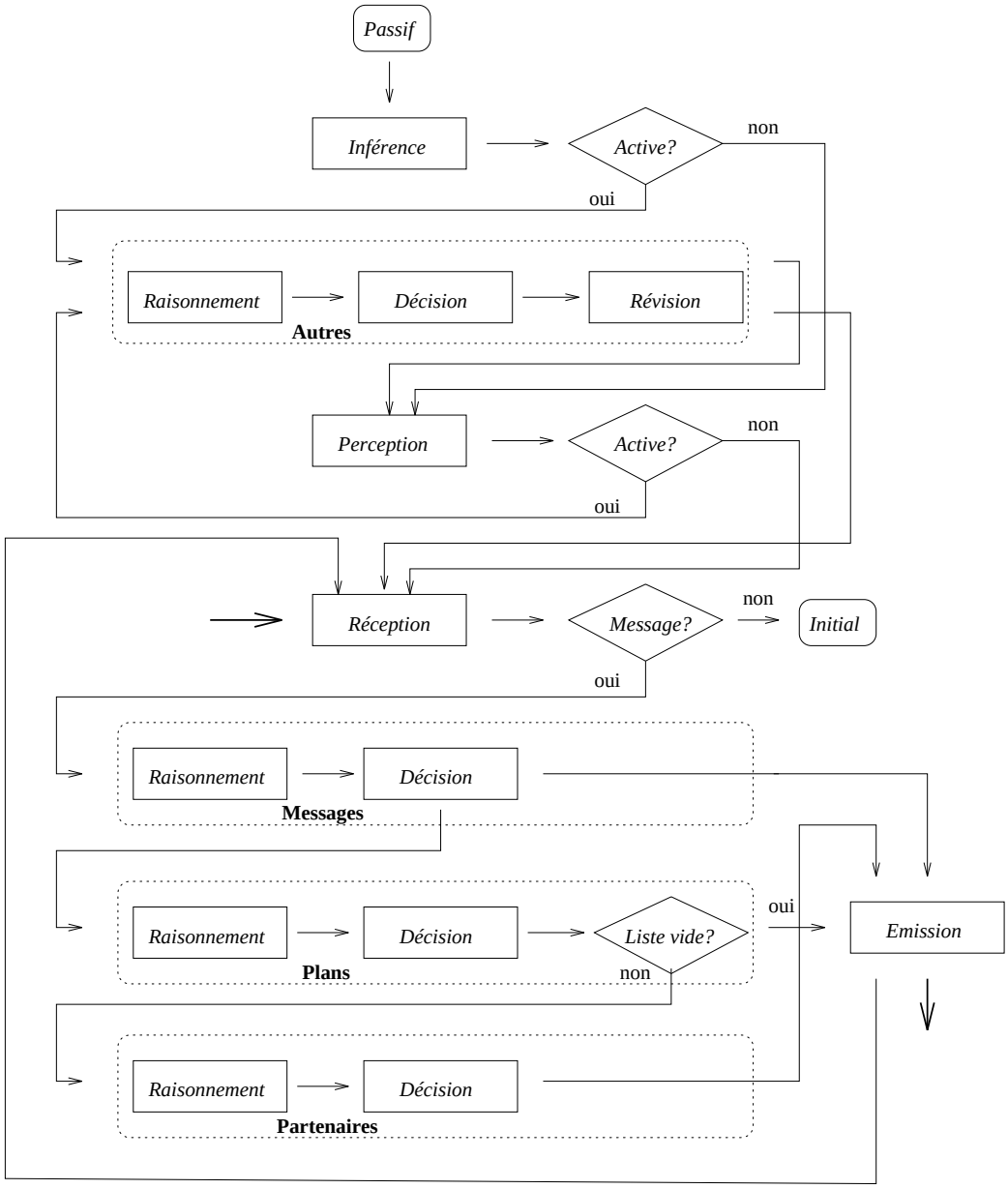


Figure 11.6 : Comportement passif

11.4 Exemples d'utilisation

Dans les sous-sections suivantes, nous présentons quelques exemples d'utilisation du système DEPINT. En particulier, nous voulons démontrer que les agents de ce système conçoivent leur organisation de façon dynamique, qu'ils sont capables aussi de s'adapter à une société ouverte et qu'ils révisent les informations qu'ils ont les uns sur les autres au fur et à mesure qu'ils interagissent.

Pour cela, nous allons reprendre l'exemple du laboratoire de recherche en électronique, introduit dans la section 10.2.2 et dont la description externe a été présentée dans l'exemple 10.5.

11.4.1 Autonomie

Supposons qu'initialement le seul agent présent dans ce laboratoire soit $ag5$. Si nous lui attribuons un comportement actif, nous aurons le résultat suivant, présenté dans l'exemple 11.1.

Exemple 11.1 : Exemple d'autonomie.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): y

===== Reasoning about goals ...

My dependence network is:
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
         |                          process_latex().
         |----------  process_latex
                    |----------  UNKNOWN
                               |----------
         |  review_oop_paper (10)
         |----------  review_oop_paper:=analyse_oop_paper().
         |                    |----------  A-AUTONOMOUS
         |                    |----------
         |  write_ss_mas_paper (30)
         |----------  write_ss_mas_paper:=write_ss_section(),
                    |                       write_mas_section(),
                    |                       process_latex().
                    |----------  write_ss_section
                               |**********  UNKNOWN
                               |          |----------
                               |  process_latex
                               |**********  UNKNOWN
                                          |----------
```

```
My current list of possible goals is :

write_mas_paper(20) non achievable
review_oop_paper(10) achievable
write_ss_mas_paper(30) non achievable

===== Deciding about goals ...

The goal selected is :

review_oop_paper (10)

===== Reasoning about plans ...

My dependence network is:
  ag5
 <ag5>
----------  review_oop_paper (10)
         |----------  review_oop_paper:=analyse_oop_paper().
                    |----------  A-AUTONOMOUS
                               |----------

My current list of possible plans is:

review_oop_paper:=analyse_oop_paper().(10) feasible

===== Deciding about plans ...

The plan selected is :

review_oop_paper:=analyse_oop_paper(). (10)

===== Reasoning about partners ...

My dependence network is:
  ag5
 <ag5>
----------  review_oop_paper (10)
         |----------  review_oop_paper:=analyse_oop_paper().
                    |----------  A-AUTONOMOUS
                               |----------

My goal situation is AUT

I do not need any actions in the commited plan

===== Deciding about partners ...
```

```
I am autonomous for the commited plan, no need of partners
```

Nous pouvons remarquer dans cet exemple que l'agent a choisi le seul but réalisable à ce moment, même si son importance est la plus petite parmi tous ses buts. Comme l'agent est autonome pour ce but, aucune proposition de coalition n'est envoyée.

11.4.2 Coalition non-réussie

Supposons que les agents $ag6$ et $ag7$ entrent dans le système, et qu'ils envoient un message de présentation[9]. En recevant de tels messages, l'agent $ag5$ met à jour sa description externe, en ajoutant les informations relatives à ces deux agents. Le nouveau résultat d'activation de l'agent $ag5$ est présenté dans l'exemple 11.2.

Exemple 11.2 : Premier exemple de proposition de coalition.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): y

===== Reasoning about goals ...

My dependence network is:
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
         |                              process_latex().
         |----------  process_latex
                    |----------  ag6
                                 |----------
                                 |  ag7
                                 |----------
         |  review_oop_paper (10)
         |----------  review_oop_paper:=analyse_oop_paper().
         |----------  A-AUTONOMOUS
                    |----------
         |  write_ss_mas_paper (30)
         |----------  write_ss_mas_paper:=write_ss_section(),
                    |                      write_mas_section(),
                    |                      process_latex().
                    |----------  write_ss_section
                               |**********  UNKNOWN
                               |          |----------
                               |  process_latex
                               |**********  ag6
                                          |----------
                                          |  ag7
                                          |----------
```

[9]Par concision, nous ne pouvons pas présenter ici la sortie de toutes les étapes du système.

```
My current list of possible goals is :

write_mas_paper(20) achievable
review_oop_paper(10) achievable
write_ss_mas_paper(30) non achievable

===== Deciding about goals ...

The goal selected is :

write_mas_paper (20)

===== Reasoning about plans ...

My dependence network is:
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
                   |                   process_latex().
                   |----------  process_latex
                              |----------  ag6
                                         |----------
                                         |  ag7
                                         |----------

My current list of possible plans is:

write_mas_paper:=write_mas_section(), process_latex().(20) feasible

===== Deciding about plans ...

The plan selected is :

write_mas_paper:=write_mas_section(), process_latex(). (20)

===== Reasoning about partners ...


My dependence network is:
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
                   |                   process_latex().
                   |----------  process_latex
                              |----------  ag6
                                         |----------
                                         |  ag7
                                         |----------
```

```
My goal situation is DEP

My needed action  is process_latex

My current list of partners is :

(ag6) UD NONE NONE
(ag7) UD NONE NONE

===== Deciding about partners ...

The partner selected is :

(ag6) UD NONE NONE

===== Sending a message ...

---> Etape : conv1 DEPINTPROPOSITION init receive "5.5.92"
---> Destinataire : you
---> Type : request
---> Ressource : matter=dec,put,moi,proposal,

There is only one possible transition in the protocol

===== Trying to receive a message ...
```

Dans cet exemple, nous pouvons constater que notre mécanisme de raisonnement social permet effectivement à un agent de s'adapter à une société ouverte: si nous comparons le résultat de l'exemple ci-dessus avec celui de l'exemple 11.1, l'agent $ag5$ a choisi maintenant un autre but à poursuivre, qui est devenu réalisable par l'arrivée des agents $ag6$ et $ag7$.

Supposons maintenant que l'agent $ag6$ ait le comportement passif présenté dans l'exemple 11.3.

Exemple 11.3 : Exemple de refus.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): n

Do you want the agent to leave the society? (y/n): n

===== Inferring properties about other agents ...

Do you want the agent to infer in this cycle? (y/n): n

===== Perceiving properties of other agents ...
```

```
Do you want the agent to perceive in this cycle? (y/n): n

===== Trying to receive a message ...

The message received is:

( PROPOSAL < ag5 polaris.imag.fr 13892 >
(write_mas_paper process_latex UD NONE NONE )

===== Reasoning about messages ...

I have received a proposal of coalition:

( PROPOSAL < ag5 polaris.imag.fr 13892 >
(write_mas_paper process_latex UD NONE NONE )

The partner has not offered any goal

===== Deciding about proposals ...

I will refuse the proposal, because there is nothing being proposed to me

===== Sending a message ...

Si error
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
---> Ressource : matter=obs,put,moi,revision,
Si !error&&!best_option
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
---> Ressource : matter=dec,put,moi,refusal,
Si !error&&best_option
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
----> Type : answer
---> Ressource : matter=dec,put,moi,acceptance,

The transition chosen is !error&&!best_option
```

L'agent $ag6$ a refusé de faire partie de la coalition, car l'agent $ag5$ ne lui a rien offert en échange[10]. Une fois reçu le message de refus, l'agent $ag5$ retire l'agent $ag6$ de la liste de partenaires possibles. Il va essayer de former une coalition avec

[10] Nous voulons faire une petite remarque ici. Dans le critère d'acceptation de partenaires présenté dans la section 7.2.2, cette situation n'est pas représentée. Ce fait se justifie par l'adoption implicite de notre principe de la non-bienveillance (P1).

l'agent $ag7$. Ce dernier va refuser aussi d'en faire partie, pour les mêmes raisons que l'agent $ag6$. Selon la description du comportement actif présenté dans la section 11.3.2, l'agent $ag5$ va recommencer à raisonner sur les plans, car la liste de partenaires possibles est maintenant vide. Comme cette liste n'a qu'un seul plan pour le but poursuivi, il va alors recommencer à raisonner sur les buts, et va essayer de poursuivre le seul but maintenant réalisable, comme dans l'exemple 11.1. Cette procédure est présentée dans l'exemple 11.4 suivant.

Exemple 11.4 : Exemple de traitement de refus.

```
===== Trying to receive a message ...

The message received is:

( REFUSAL < ag6 polaris.imag.fr 13893 >  )

===== Reasoning about messages ...

The partner has refused to form a coalition

===== Reasoning about partners ...

The commited partner has refused to form a coalition

Removing the partner (ag6) from the list of possible partners
My current list of partners is :

(ag7) UD NONE NONE

===== Deciding about partners ...

The partner selected is :

(ag7) UD NONE NONE

===== Sending a message ...

---> Etape : conv1 DEPINTPROPOSITION init receive "5.5.92"
---> Destinataire : you
---> Type : request
---> Ressource : matter=dec put moi proposal

There is only one possible transition in the protocol

===== Trying to receive a message ...

The message received is:

( REFUSAL < ag7 polaris.imag.fr 13894 >  )
```

```
===== Reasoning about messages ...

The partner has refused to form a coalition

===== Reasoning about partners ...

The commited partner has refused to form a coalition

Removing the partner (ag7) from the list of possible partners

My current list of possible partners is empty

===== Deciding about partners ...

There are no more partners for the commited plan

===== Reasoning about plans ...

The commited plan is no longer feasible

Removing the plan

write_mas_paper:=write_mas_section(), process_latex().

from the list of possible plans

My current list of possible plans is empty

===== Deciding about plans ...

I do not have any more plans to achieve the commited goal

===== Reasoning about goals ...

The commited goal is no longer achievable

Removing the goal write_mas_paper from the list of possible goals

My current list of possible goals is :

review_oop_paper(10) achievable

===== Deciding about goals ...

The goal selected is :

review_oop_paper (10)
```

11.4.3 Coalition réussie

Supposons maintenant qu'un nouvel agent, $ag8$, entre dans la société. Cet agent a le but *write_mas_paper* à atteindre et il peut aussi entreprendre l'action *process_latex*. Néanmoins, l'agent $ag8$ n'a aucun plan pour atteindre son but.

Une fois le protocole de présentation déroulé, le nouveau comportement de l'agent $ag5$ est présenté dans l'exemple 11.5.

Exemple 11.5 : Deuxième exemple de proposition de coalition.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): y

===== Reasoning about goals ...

My dependence network is:
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
         |         |                   process_latex().
         |         |----------  process_latex
         |                    |----------  ag6
         |                                 |----------
         |                                 |  ag7
         |                                 |----------
         |                                 |  ag8
         |                                 |----------
         |  review_oop_paper (10)
         |----------  review_oop_paper:=analyse_oop_paper().
         |         |----------  A-AUTONOMOUS
         |                    |----------
         |  write_ss_mas_paper (30)
         |----------  write_ss_mas_paper:=write_ss_section(),
                   |                       write_mas_section(),
                   |                       process_latex().
                   |----------  write_ss_section
                              |**********  UNKNOWN
                              |         |----------
                              |  process_latex
                              |**********  ag6
                                         |----------
                                         |  ag7
                                         |----------
                                         |  ag8
                                         |----------
```

```
My current list of possible goals is :

write_mas_paper(20) achievable
review_oop_paper(10) achievable
write_ss_mas_paper(30) non achievable


===== Deciding about goals ...


The goal selected is :

write_mas_paper (20)


===== Reasoning about plans ...


My dependence network is:
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
                    |                   process_latex().
                    |----------  process_latex
                               |----------  ag6
                                           |----------
                                           |  ag7
                                           |----------
                                           |  ag8
                                           |----------

My current list of possible plans is:

write_mas_paper:=write_mas_section(), process_latex().(20) feasible


===== Deciding about plans ...


The plan selected is :

write_mas_paper:=write_mas_section(), process_latex(). (20)


===== Reasoning about partners ...


My dependence network is:
```

```
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
                    |                  process_latex().
                    |----------  process_latex
                               |----------  ag6
                                          |----------
                                          |  ag7
                                          |----------
                                          |  ag8
                                          |----------


My goal situation is DEP

My needed action  is process_latex

My current list of partners is :

(ag6) UD NONE NONE
(ag7) UD NONE NONE
(ag8) LBMD write_mas_paper write_mas_section

===== Deciding about partners ...

The partner selected is :

(ag8) LBMD write_mas_paper write_mas_section

===== Sending a message ...

---> Etape : conv1 DEPINTPROPOSITION init receive "5.5.92"
---> Destinataire : you
---> Type : request
---> Ressource : matter=dec put moi proposal

There is only one possible transition in the protocol

===== Trying to receive a message ...
```

Nous pouvons noter que maintenant l'agent $ag5$ choisit comme partenaire l'agent $ag8$, car selon le critère de choix de partenaires présenté dans la section 7.1.4 une $LBMD$ est une situation meilleure qu'une UD.

Le comportement de l'agent $ag8$ est présenté dans l'exemple 11.6.

Exemple 11.6 : Exemple d'acceptation.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): n

Do you want the agent to leave the society? (y/n): n

===== Inferring properties about other agents ...

Do you want the agent to infer in this cycle? (y/n): n

===== Perceiving properties of other agents ...

Do you want the agent to perceive in this cycle? (y/n): n

===== Trying to receive a message ...

The message received is:

( PROPOSAL < ag5 polaris.imag.fr 13892 >
(write_mas_paper process_latex LBMD write_mas_paper write_mas_section )

===== Reasoning about messages ...

I have received a proposal of coalition:

( PROPOSAL < ag5 polaris.imag.fr 13892 >
(write_mas_paper process_latex LBMD write_mas_paper write_mas_section )

My dependence network is:
  ag8
 <ag8>
----------  write_mas_paper (10)
         |----------  NO-PLANS
                    |----------


My goal situation is NP

===== Deciding about proposals ...

I will accept the proposal, because I do not have a plan for this goal

===== Sending a message ...

Si error
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
```

```
---> Ressource : matter=obs,put,moi,revision,
Si !error&&!best_option
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
---> Ressource : matter=dec,put,moi,refusal,
Si !error&&best_option
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
---> Ressource : matter=dec,put,moi,acceptance,

The transition chosen is !error&&best_option
```

Comme nous l'avons prevu dans notre critère d'acceptation, la réponse est une acceptation, car la situation de but de l'agent $ag8$ est NP. Le déroulement du traitement du message chez l'agent $ag5$ est montré dans l'exemple 11.7.

Exemple 11.7 : Exemple de traitement d'acceptation.

```
===== Trying to receive a message ...

The message received is:

( ACCEPTANCE < ag8 polaris.imag.fr 13895 >  )

===== Reasoning about messages ...

*** The partner has accepted to form a coalition ***
```

11.4.4 Révision de croyances

Supposons finalement que les agents $ag6$, $ag7$ et $ag8$ quittent la société et qu'un nouvel agent, $ag9$, y entre. Cet agent a le but *write_ss_mas_paper* à atteindre, mais il n'a aucun plan pour ce but et il ne peut entreprendre aucune action non plus.

Supposons que l'agent $ag5$ infère que l'agent $ag9$ est capable d'entreprendre les actions *write_ss_section* et *process_latex*, comme le montre l'exemple 11.8 pour l'action *write_ss_section*.

Exemple 11.8 : Exemple d'inférence.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): n
```

```
Do you want the agent to leave the society? (y/n): n

===== Inferring properties about other agents ...

Do you want the agent to infer in this cycle? (y/n): y

Type the name of the agent to be selected: ag9

Inference may be about goals, actions, resources or plans

Type the selected option (G/A/R/P): a

The current actions of agent < ag9 polaris.imag.fr 14015 > are:


Entries may be inserted or removed

Type the selected option (I/R): i

Type the INCOMPLETE ACTION: write_ss_section

Type the cost  of the ACTION: 10

===== Reasoning about the others ...

I must revise the following information:

( (ag9) ACTION INCOMPLETE write_ss_section )

===== Deciding about the others ...

Incomplete information is always updated

===== Revising information about the Others ...

Updating the external description

Action write_ss_section was included

in the external description entry of agent

< ag9 polaris.imag.fr 14015 >
```

Maintenant, l'agent $ag5$ va choisir de poursuivre le but le plus important pour lui, car celui-ci est devenu réalisable, comme le montre l'exemple 11.9.

Exemple 11.9 : Troisième exemple de proposition de coalition.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): y

===== Reasoning about goals ...

My dependence network is:
  ag5
 <ag5>
----------  write_mas_paper (20)
         |----------  write_mas_paper:=write_mas_section(),
         |         |                  process_latex().
         |         |----------  process_latex
         |                    |----------  ag9
         |                               |----------
         |  review_oop_paper (10)
         |----------  review_oop_paper:=analyse_oop_paper().
         |         |----------  A-AUTONOMOUS
         |                    |----------
         |  write_ss_mas_paper (30)
         |----------  write_ss_mas_paper:=write_ss_section(),
                    |                     write_mas_section(),
                    |                     process_latex().
                    |----------  write_ss_section
                               |**********  ag9
                               |         |----------
                               |  process_latex
                               |**********  ag9
                                         |----------


My current list of possible goals is :

write_mas_paper(20) achievable
review_oop_paper(10) achievable
write_ss_mas_paper(30) achievable

===== Deciding about goals ...

The goal selected is :

write_ss_mas_paper (30)

===== Reasoning about plans ...

My dependence network is:
```

```
  ag5
 <ag5>
----------  write_ss_mas_paper (30)
         |----------  write_ss_mas_paper:=write_ss_section(),
                   |                      write_mas_section(),
                   |                      process_latex().
                   |----------  write_ss_section
                              |**********  ag9
                              |          |----------
                              |  process_latex
                              |**********  ag9
                                         |----------

My current list of possible plans is:

write_ss_mas_paper:=write_ss_section(), write_mas_section(),
                    process_latex().(30) feasible

===== Deciding about plans ...

The plan selected is :

write_ss_mas_paper:=write_ss_section(), write_mas_section(),
                    process_latex(). (30)

===== Reasoning about partners ...

My dependence network is:
  ag5
 <ag5>
----------  write_ss_mas_paper (30)
         |----------  write_ss_mas_paper:=write_ss_section(),
                   |                      write_mas_section(),
                   |                      process_latex().
                   |----------  write_ss_section
                              |**********  ag9
                              |          |----------
                              |  process_latex
                              |**********  ag9
                                         |----------

My goal situation is DEP

My needed action  is write_ss_section

My current list of partners is :

(ag9) LBMD write_ss_mas_paper write_mas_section
```

```
===== Deciding about partners ...

The partner selected is :

(ag9) LBMD write_ss_mas_paper write_mas_section

==== Sending a message ...

---> Etape : conv1 DEPINTPROPOSITION init receive "5.5.92"
---> Destinataire : you
---> Type : request
---> Ressource : matter=dec,put,moi,proposal,

There is only one possible transition in the protocol

===== Trying to receive a message ...
```

Lorsque l'agent *ag*9 reçoit cette proposition, elle se rend compte que l'agent *ag*5 a une croyance fausse relative à l'action *write_ss_section*. Ainsi, cet agent va lui demander d'entamer une révision de croyances, comme le montre l'exemple 11.10.

Exemple 11.10 : Exemple de détection d'inconsistance.

```
===== Initial state ...

Do you want the agent to be active in this cycle? (y/n): n

Do you want the agent to leave the society? (y/n): n

===== Inferring properties about other agents ...

Do you want the agent to infer in this cycle? (y/n): n

===== Perceiving properties of other agents ...

Do you want the agent to perceive in this cycle? (y/n): n

===== Trying to receive a message ...

The message received is:

( PROPOSAL < ag5 polaris.imag.fr 13892 > (write_ss_mas_paper
write_ss_section LBMD write_ss_mas_paper write_mas_section )

===== Reasoning about messages ...

I have received a proposal of coalition:
```

```
( PROPOSAL < ag5 polaris.imag.fr 13892 > (write_ss_mas_paper
write_ss_section LBMD write_ss_mas_paper write_mas_section )

My dependence network is:
  ag9
 <ag9>
----------  write_ss_mas_paper (10)
         |----------  NO-PLANS
                    |----------

My goal situation is NP

===== Deciding about proposals ...

I will refuse the proposal, because I do not have the needed action

===== Sending a message ...

Si error
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
---> Ressource : matter=obs,put,moi,revision,
Si !error&&!best_option
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
---> Ressource : matter=dec,put,moi,refusal,
Si !error&&best_option
---> Etape : conv1 DEPINTPROPOSITION init end "5.5.92"
---> Destinataire : you
---> Type : answer
---> Ressource : matter=dec,put,moi,acceptance,

The transition chosen is error
```

En adoptant toujours notre critère de choix de contexte défini dans la section 8.5.2, l'agent $ag5$ va préférer enlever l'action *write_ss_section* de l'entrée de description externe de l'agent $ag9$, car la source de la nouvelle information est plus fiable que la source antérieure. Cette procédure est illustrée dans l'exemple 11.11.

Exemple 11.11 : Exemple de révision de croyances.

```
===== Trying to receive a message ...

The message received is:
```

```
( REVISION < ag9 polaris.imag.fr 14015 >
( (ag9) ACTION INCORRECT write_ss_section )

===== Reasoning about messages ...

The partner has asked me to do a revision

===== Reasoning about the others ...

I must revise the following information:

( (ag9) ACTION INCORRECT write_ss_section )

===== Deciding about the others ...

Topic is (ag9)

Previous source was: <Inference>

New source is <Communication(ag9)>

New source is preferable

===== Revising information about the Others ...

Updating the external description

Action write_ss_section was removed from
the external description entry of agent
< ag9 polaris.imag.fr 14015 >
```

11.5 Discussion

Jusqu'à présent, nous ne connaissons pas d'autres systèmes qui utilisent la notion de dépendance sociale pour former dynamiquement des coalitions entre agents dans un cadre d'un SMA ouvert. La plupart des méthodes de résolution coopérative de problèmes que nous trouvons dans la littérature, comme par exemple celle présentée dans [WJ94], ne s'adressent qu'à un niveau formel.

Si nous comparons la méthode proposée dans ce système avec le réseau contractuel ("contract net") [Smi80], nous croyons que dans notre approche *le flux global de communication de la société est diminué.* Même si dans le contexte du système DEPINT chaque agent diffuse un message à tous les autres au moment de son introduction, cela n'est fait qu'une seule fois, et les agents peuvent prendre en compte les informations qu'ils possèdent sur les autres afin de limiter le nombre d'agents auxquels ils envoient des propositions de formation de coalition.

Néanmoins, en ce qui concerne les fonctionnalités de ce système, nous envisageons certaines améliorations :

- nous pouvons *mieux exploiter certains aspects* de notre modèle d'agent présenté dans la section 3.4. Par exemple, le fait qu'un agent puisse inférer des nouvelles informations sur les autres à partir des résultats générés par son mécanisme de perception ou d'inférence sur le domaine n'a pas été suffisamment exploité au sein de ce système ;

- dans le cadre de ce système, l'activation des mécanismes internes est faite de façon séquentielle. Nous pouvons bien envisager de les *activer de façon concurrente*, en permettant l'application de tactiques de contrôle plus flexibles ;

- le protocole de présentation doit être complété, afin de permettre aux agents d'envoyer effectivement leurs entrées de description externe lorsqu'ils entrent dans la société ;

- le système doit aussi être étendu pour pouvoir traiter les cas de dépendances multi-partites.

Les points abordés ci-dessus font partie de nos perspectives, présentées dans la section 12.2.

Chapitre 12

Conclusion

Dans ce chapitre, nous présentons la section 12.1 une synthèse des divers problèmes abordés dans cette thèse, ainsi que les solutions que nous avons proposé pour les résoudre. Ensuite, nous présentons dans la section 12.2 nos perspectives de recherche. Nous abordons aussi dans la section 12.3 une exploitation possible de notre modèle qui n'a pas été traité dans le cadre de cette thèse. Il s'agit de l'utiliser comme un outil formel pour décrire et concevoir de façon statique les organisations d'une société d'agents.

12.1 Problèmes abordés

Dans cette thèse, nous avons proposé le modèle d'un mécanisme de raisonnement social fondé sur la théorie de la dépendance. Nous avons montré que ce modèle permet à un agent de raisonner sur autrui, et que un tel type de raisonnement est important dans un contexte de SMA ouvert.

Nous avons d'abord montré que dans le cadre de tels systèmes, l'organisation des agents ne peut pas être spécifiée de façon statique, pendant la phase de conception. Nous avons adopté ainsi une méthode de résolution coopérative de problèmes fondée sur la formation dynamique de coalitions, et nous avons montré que *notre mécanisme de raisonnement social peut être utilisé comme une base pour la formation de coalitions*. En ne supposant pas que les agents soient toujours bienveillants, nous avons proposé un critère de choix partenaires basé sur les situations de dépendance qui permet aux agents d'évaluer la possibilité des autres agents à accepter ses propositions de formation de coalitions. En outre, nous avons indiqué que dans un contexte d'agent bienveillants, nous pouvons relaxer ce critère, en n'utilisant que les relations de dépendance. Nous avons aussi montré que notre modèle permet à un agent de faire évoluer ses choix de partenaires, en prenant compte de l'aspect dynamique de la société d'agents.

Comme dans ces systèmes des fonctionnalités de résolution peuvent être créées ou détruites, de façon dynamique, par l'entrée ou la sortie de nouveaux agents, nous avons montré que notre mécanisme permet aussi aux agents de *s'adapter* à

ces changements, en les aidant à évaluer à tout moment si ses buts sont réalisables et si ses plans sont exécutables.

Enfin, comme dans de tels systèmes il est fortement improbable que tous les agents aient des informations correctes et complètes les uns sur les autres à tout moment, nous avons montré comment notre modèle permet à un agent de détecter si sa représentation des autres est incomplète ou incorrecte. Nous avons ensuite proposé un critère de choix de contexte, fondé sur les notions de source et de thème d'information, pour la *révision de la représentation des autres.*

Nous avons développé deux systèmes, le simulateur DEPNET et le système DEPINT, qui illustrent que notre modèle peut être utilisé selon deux perspectives scientifiques diverses. Dans un cadre de simulation sociale, il fournit un cadre pour analyser et prédire des interactions sociales et pour l'analyse du pouvoir social des agents. En ce qui concerne une perspective de résolution de problèmes, notre modèle peut être utilisé pour concevoir dynamiquement l'organisation d'agents dans un contexte de SMA ouvert.

A notre connaissance, aucun travail n'a encore abordé les sujets suivants, ceux-ci constituant notre principale contribution :

- un modèle de résolution coopérative de problèmes qui permet une révision incrémentale de croyances comme un effet de la procédure de formation de coalitions ;

- la prise en compte de la réalisabilité des buts et de la l'exécutabilité des plans dans un contexte de SMA ouvert ;

La conséquence la plus évidente d'un tel mécanisme est la *diminution du flux global de communication de la société.* Même si dans le contexte du système DEPINT chaque agent diffuse un message à tous les autres au moment de son introduction, cela n'est fait qu'une seule fois. Comme les autres peuvent prendre en compte cette information, il n'y a plus besoin pour un agent d'envoyer un message à tous les autres lorsqu'il a besoin d'une action ou ressource, comme dans le cas du réseau contractuel ("contract net") [Smi80], puisqu'a priori il connaît les agents auxquels il doit s'adresser.

12.2 Perspectives

Dans les sections suivantes, nous énumérons les possibles évolutions que nous envisageons pour les différents aspects abordés dans cette thèse.

12.2.1 Modèle de raisonnement social

Par rapport au modèle proposé dans le chapitre 5, nous voulons l'améliorer en ce qui concerne les aspects suivants :

- nous voulons proposer un modèle pour la *quantification de la dépendance*. Ce modèle peut être basé, par exemple, sur l'importance des buts, le nombre d'actions/ressources nécessaires à l'exécution d'un plan et le nombre d'agents capables d'entreprendre une certaine action/contrôler une certaine ressource. Comme nous avons illustré dans l'exemple de la fédération de laboratoires de recherche présenté dans le chapitre 10, un agent intuitivement dépend moins d'un autre si il existe d'autres agents qui peuvent entreprendre la même action/contrôler la même ressource. Nous croyons que cette évolution aura plutôt un effet sur la perspective SS de nos recherches, car à partir de cette quantification nous pouvons envisager une théorie plus adéquate pour le pouvoir social ;

- nous voulons aussi étendre la taxonomie de situations de dépendance, afin de prendre en compte les dépendances de ressources. Celles-ci n'ont pas été prises en compte dans le cadre de ce travail, comme nous l'avons explicité dans la section 5.3.8. Ainsi, il deviendra possible de représenter explicitement dans notre modèle les dépendances externes et transitives, telles que nous les avons décrites dans la section 4.6.3 ;

- enfin, nous voulons exploiter les interférences sociales négatives proposées dans la section 4.3.1, comme la compétition et le conflit. Nous croyons qu'un agent "intelligent" doit aussi pouvoir raisonner et agir dans un tel contexte.

12.2.2 Critères de choix de buts, plans et partenaires

Par rapport aux divers critères de choix présentés dans le chapitres 6 et 7, nous envisageons les évolutions suivantes :

- nous croyons que les critères de choix et d'acceptation de partenaires que nous avons proposé dans la section 7.1.4 peuvent être modifiés de façon à pouvoir caractériser des différents types d'agent. Par exemple, selon le critère de choix de partenaires que nous avons défini, un agent préfère toujours travailler seul que coopérer. Néanmoins, d'autres agents peuvent préférer toujours coopérer que travailler seuls, et de cette façon l'adoption de différents critères pour différents agents nous permet d'avoir un cadre de modélisation cognitive plus riche ;

- nous voulons tester si l'ordre d'activation de ces critères a un effet significatif sur les divers choix d'un agent. Autrement dit, cela veut dire ne plus considérer qu'un agent va toujours choisir un but, après un plan et seulement après les partenaires, comme nous l'avons adopté dans cette thèse (cf. section 6.1). Nous considérons qu'il existe au moins deux situations intéressantes à analyser :

  - ⋆ si on supprime la restriction introduite dans la section 5.1.1, où nous avons supposé que tous les agents évaluent de la même façon les coûts des actions et des ressources, il devient clair que le choix de plans doit être fait en considérant les partenaires possibles, puisque le coût du plan va dépendre de ceux-ci ;
  - ⋆ puisque le critère de choix de partenaires n'est pas quantifié, nous n'avons pu le combiner avec l'importance de buts et le coût des plans. Nous croyons que cet aspect est important : Est-ce qu'il est plus rationnel pour un agent de choisir un but dont l'importance est 100 et dont sa meilleure situation de dépendance est une UD ou bien de choisir un but dont l'importance est 80, mais dont par contre il existe un partenaire envers lequel il a déduit une MBMD ? La même question peut être analysé en ce qui concerne les coûts des plans ;

- nous n'avons pas considéré dans nos critères des notions dynamiques, tels que l'engagement. Par exemple, un agent peut bien refuser de faire partie d'une coalition parce que il est déjà engagé dans une autre pour atteindre le même but. Cet aspect doit être prise en compte dans nos travaux futurs.

12.2.3 Révision de croyances

En ce qui concerne la procédure de révision de croyances, nous voulons aborder les aspects suivants :

- selon notre modèle, lorsqu'un agent détecte qu'il a une croyance fausse relative à un autre agent (car ce dernier lui a communiqué ce fait), il révise sa représentation relative à cet agent. Néanmoins, il pourrait aussi raisonner sur les causes qui l'ont amené à inférer cette proposition. Si par exemple celle-ci a été acquise par perception, il pourrait bien mettre en cause la correction de ce mécanisme interne. Si, par contre, cette proposition a été acquise par inférence, nous pouvons envisager un protocole d'interaction pour lui permettre de découvrir laquelle parmi ses règles de déduction n'est pas applicable. De cette façon, un agent peut avoir un mécanisme interne de haut-niveau qui lui permet de raisonner sur ses propres mécanismes internes ;

- si nous ne considérons plus le principe d'auto-connaissance (P3) comme valable, nous avons une autre situation très intéressante : un agent peut apprendre avec d'autres agents. Considérons un exemple d'un robot qui sait peindre des murs, mais qui ne sait pas que les mêmes actions élémentaires lui permettent aussi de nettoyer des murs. Si un deuxième robot a une règle du type $is_a(i, paint) \Rightarrow is_a(i, clean)$, lors d'une procédure de formation de coalition ce deuxième pourrait enseigner au premier ce fait. Dans un certain sens, une telle procédure est similaire à celle que les chercheurs dans la communauté d'apprentissage appellent apprentissage par instruction

[MCM83]. Cet aspect a été abordé déjà dans [BDB92] sous le nom de problème d'incohérence, lorsqu'on propose un protocole d'introduction d'agents dans une société ;

- enfin, nous pouvons aussi ne plus considérer le principe de sincérité (P2) comme valable. Selon une perspective SS, il est intéressant de pouvoir modéliser la situation où un agent communique de façon délibérée aux autres qu'il est capable d'entreprendre une certaine action lorsqu'il ne l'est pas vraiment. Ce comportement se justifie comme un moyen d'obtenir du pouvoir social sur les autres, comme nous l'avons analysé dans le chapitre 4. Nous croyons que dans ce cas, les situations de dépendance peuvent être prises en compte dans un critère de choix de contexte. Ainsi, un agent i n'accepterais pas comme toujours vraies les informations que l'agent j lui a envoyées sur lui-même lorsque, par exemple, il a inféré une *MBRD* envers lui.

12.2.4 Implémentation et applications

En ce qui concerne les applications développées dans le cadre de cette thèse, nous voulons les améliorer dans les aspects suivants :

- nous voulons implémenter une interface plus conviviale pour le simulateur DEPNET, en utilisant des ressources graphiques plus sophistiquées. Nous voulons aussi proposer quelques fonctions additionnelles qui permettent une évaluation plus précise du pouvoir social des agents, dont une première tentative a été présenté dans [CC95] ;

- en ce qui concerne le système DEPINT, nous voulons implémenter les mécanismes internes de notre modèle d'agent de façon distribuée, en utilisant le système DPSK+P [CS92]. De cette façon, nous croyons que nous aurons la possibilité de tester différentes tactiques d'activation de ses mécanismes internes. Nous avons aussi l'intention de l'étendre pour pouvoir traiter les cas de dépendances multi-partites. A court terme, nous envisageons aussi d'implémenter de façon complète le protocole de présentation.

12.3 Conception statique des organisations

Même si nous n'avons pas exploité cette voie dans cette thèse, nous croyons que le modèle présenté dans le chapitre 5 peut être utilisé en tant qu'outil formel pour décrire et concevoir de façon statique des organisations d'agents. Dans notre contexte de recherche, les agents eux-mêmes ont une représentation *subjective* de leurs dépendances, c'est-à-dire qu'il existe une représentation mentale interne qui représente et exploite cette information. C'est à cet aspect-là auquel nous nous intéressons plus particulièrement dans notre recherche. Néanmoins, nous croyons que le modèle de dépendances est suffisamment riche pour être utilisé dans un

deuxième cadre, où les agents[1] n'ont aucune représentation interne explicite de ses dépendances. Dans ce dernier cas, les dépendances (représentant en quelque sorte l'organisation de la société) sont implicitement prises en compte par le concepteur lorsqu'il conçoit le système.

Puisque notre objectif scientifique dans cette thèse n'était pas celui de proposer un modèle pour la conception statique des organisations, nous n'avons pas fait une étude comparative détaillée avec d'autres modèles de ce genre. Néanmoins, nous en avons analysé quelques uns, afin de détecter d'éventuelles différences par rapport aux notions de base utilisées.

Dans [YM93], un modèle de dépendances entre acteurs est présenté dans un cadre de modélisation des entreprises ("business reengineering"). Dans ce modèle, l'auteur propose quatre types de dépendances : de ressources, de tâches, de buts et de ce qu'il appelle "soft-goals". Une dépendance de but exprime le fait que l'agent qui dépend de l'autre ne s'intéresse par la manière dont le but est atteint (ce qui sert à modéliser l'acquisition de biens). Une dépendance de tâches exprime le fait que l'agent censé l'exécuter n'est pas au courant du but auquel cette tâche doit servir (ce qui sert à modéliser les relations du type maître-esclave). Un "soft-goal" est défini comme un but où l'agent dépendant spécifie les moyens qui doivent être utilisés pour l'atteindre : il existe une espèce de négociation de plan en cours d'exécution. Finalement, une dépendance de ressource a le même sens que dans notre travail. Par rapport à notre modèle, nous croyons que notre notion de dépendance d'action correspond à une dépendance de "soft-goals", car pendant la formation de coalitions, les buts sont communiqués ainsi que les actions requises.

Dans la méthode CASSIOPÉE, inspirée par nos travaux [CCZ94, page 4], le concepteur d'un SMA définit les comportements élémentaires qui doivent avoir lieu pendant l'activité de résolution. Ensuite, on étudie l'influence de ces comportements les uns sur les autres, afin de proposer les possibles regroupements de ces derniers au sein d'un agent. De cette façon, on arrive à limiter le flux de communication globale du système par la minimisation de la distribution des comportements qui ont de l'influence l'un sur l'autre entre plusieurs agents. Nous croyons que cette méthode est très intéressante, même si les agents, dans ce contexte, ne sont pas autonomes dans le sens qu'on a adopté dans la section 4.2.

Cependant, aucune de ces deux méthodes n'a été implémentée de façon à fournir au concepteur un outil informatique pour cette tâche. Nous croyons que le simulateur DEPNET peut être utilisé aussi dans ce contexte. Nous avons même réalisé quelques études préliminaires concernant cet aspect, qui sont présentées dans les annexes B et C, où nous illustrons son utilisation, ainsi que celle de notre modèle en tant qu'outil formel, pour décrire les organisations de deux systèmes, le système EB [SJ92] et le système MAVI [BD94c].

[1] Une fois de plus, l'utilisation du terme agent est un abus de langage dans ce contexte, mais nous tenons toujours à l'utiliser par une question d'uniformisation de rédaction.

Annexes

Annexe A

Preuves

Dans cette annexe, nous présentons les preuves formelles de certains théorèmes énoncés dans le chapitre 8, en utilisant les prédicats qui ont été définis ainsi que ceux définis dans le chapitre 5.

A.1 Remarques sur les preuves

Dans les preuves qui suivent, nous utilisons les méta-symboles suivants :

- D pour faire référence aux définitions introduites dans le chapitre 5
- T ou C pour faire référence aux théorèmes et corollaires
- H pour désigner l'hypothèse
- E pour marquer l'élimination d'un connectif où d'un quantificateur (ex : $\wedge$E)
- I pour désigner l'introduction d'un connectif où d'un quantificateur (ex : $\exists$I)
- R pour marquer la distribution d'un quantificateur (ex : $\exists$R)
- SU pour signifier la substitution de deux formules logiquement équivalentes
- MP pour le Modus Ponens
- DM pour les lois de De Morgan
- CM pour la commutativité
- IM pour la définition de l'implication
- $(x \mid y)$ pour signifier que le terme x est substitué par le terme y

Rappelons que nous utilisons des variables et constantes *typées*, avec la convention suivante :

- $\{i,\ j,\ k\}$: variables qui désignent des agents ;
- $\{g,\ g'\}$: variables qui désignent des buts ;
- $\{p,\ p'\}$: variables qui désignent des plans ;
- $\{\pi,\ \pi'\}$: constantes qui désignent des plans ;
- $\{a,\ a'\}$: variables qui désignent des actions ;
- $\{\alpha,\ \alpha'\}$: constantes qui désignent des actions ;
- $\{r,\ r'\}$: variables qui désignent des ressources ;
- $\{\rho,\ \rho'\}$: constantes qui désignent des ressources.

L'indice des prédicats dans le langage interne n'a aucun lien avec les variables qui y apparaissent. Par exemple, $needs_a(i,p,a)$ signifie que l'agent i a besoin de l'action a (qu'il est incapable d'entreprendre) qui est utilisée dans un plan p. L'indice différencie le prédicat $need_a$ du prédicat $need_r$.

Par ailleurs, dans les théorèmes et les corollaires concernant le langage externe, les indices des opérateurs modaux $\mathbf{B_i}$ et $\mathbf{B_j}$ sont liés aux agents sur lesquels l'hypothèse de compatibilité des descriptions externes est faite. Par exemple, dans une formule du genre $Ext_c(i,j) \wedge \mathbf{B_i}\ \phi \Rightarrow \mathbf{B_j}\ \phi$, les indices i et j sont liés aux variables i et j qui apparaissent dans le prédicat Ext_c.

Nous supposons que le langage interne est complet et correct, c'est-à-dire que pour n'importe quelle formule ϕ soit l'agent déduit ϕ soit il déduit $\neg\phi$, mais jamais les deux en même temps. Par conséquent, dans le langage externe nous aurons que la formule suivante est valable :

$$\neg\mathbf{B_i}\ \phi \quad \Leftrightarrow \quad \mathbf{B_i}\ \neg\phi \tag{A.1}$$

Enfin, nous supposons aussi que les notions de base introduites dans la section 5.3.1 sont interprétées de façon analogue pour tous les agents. Autrement dit si un agent i croît, par exemple, qu'un plan p atteint le but g, un autre agent j y croira aussi. En utilisant les opérateurs modaux définis ci-dessus, nous pouvons exprimer ces faits de façon formelle par les prédicats suivants :

$$\mathbf{B_i}\ achieves_g(p,g) \quad \Leftrightarrow \quad \mathbf{B_j}\ achieves_g(p,g) \tag{A.2}$$
$$\mathbf{B_i}\ uses_a(p,a) \quad \Leftrightarrow \quad \mathbf{B_j}\ uses_a(p,a) \tag{A.3}$$
$$\mathbf{B_i}\ uses_r(p,r) \quad \Leftrightarrow \quad \mathbf{B_j}\ uses_r(p,r) \tag{A.4}$$
$$\mathbf{B_i}\ diff_g(g,g') \quad \Leftrightarrow \quad \mathbf{B_j}\ diff_g(g,g') \tag{A.5}$$

A.2 Langage interne

Théorème A.2.1. $dep\_on_a(i,j,g,k) \Rightarrow \exists a\ (\neg is_a(i,a) \wedge is_a(j,a))$

Preuve :

1. $dep\_on_a(i,j,g,k)$	H
2. $dep_a(i,j,k) \wedge \exists p\ dep\_plan_a(i,j,g,p,k)$	MP 1 D5.34
3. $\exists p\ dep\_plan_a(i,j,g,p,k)$	$\wedge$E 2
4. $dep\_plan_a(i,j,g,\pi,k)$	$\exists$E 3
5. $is\_plan_g(k,g,\pi) \wedge \exists a\ (needs_a(i,\pi,a) \wedge is_a(j,a))$	MP 4 D5.27
6. $\exists a\ (needs_a(i,\pi,a) \wedge is_a(j,a)))$	$\wedge$E 5
7. $needs_a(i,\pi,\alpha) \wedge is_a(j,\alpha)$	$\exists$E 6
8. $is_a(j,\alpha)$	$\wedge$E 7
9. $needs_a(i,\pi,\alpha)$	$\wedge$E 7
10. $uses_a(\pi,\alpha) \wedge \neg is_a(i,\alpha)$	MP 9 D5.21 $(p \mid \pi)$ $(a \mid \alpha)$
11. $\neg is_a(i,\alpha)$	$\wedge$E 10
12. $\neg is_a(i,\alpha) \wedge is_a(j,\alpha)$	$\wedge$I 11 8
13. $\exists a(\neg is_a(i,a) \wedge is_a(j,a))$	$\exists$I 12

□

Corollaire A.2.1. $\forall a\ (is_a(i,a) \vee \neg is_a(j,a)) \Rightarrow \neg dep\_on_a(i,j,g,k)$

Preuve :

1. $dep\_on_a(i,j,g,k) \Rightarrow \exists a\ (\neg is_a(i,a) \wedge is_a(j,a))$	TA.2.1
2. $\neg dep\_on_a(i,j,g,k) \vee \exists a\ (\neg is_a(i,a) \wedge is_a(j,a))$	IM 1
3. $\neg dep\_on_a(i,j,g,k) \vee \neg\neg\exists a\ (\neg is_a(i,a) \wedge is_a(j,a))$	$\neg$I 2
4. $\neg dep\_on_a(i,j,g,k) \vee \neg\forall a\ (\neg\neg is_a(i,a) \vee \neg is_a(j,a))$	DM 3
5. $\neg\forall a\ (\neg\neg is_a(i,a) \vee \neg is_a(j,a)) \vee \neg dep\_on_a(i,j,g,k)$	CM 4
6. $\neg\forall a\ (is_a(i,a) \vee \neg is_a(j,a)) \vee \neg dep\_on_a(i,j,g,k)$	$\neg$E 5
7. $\forall a\ (is_a(i,a) \vee \neg is_a(j,a)) \Rightarrow \neg dep\_on_a(i,j,g,k)$	IM 6

□

A.3 Hypothèse de commutativité du langage externe

Nous allons prouver des théorèmes de la forme suivante :

$$\begin{aligned} Comp_{\psi}(i,j,k) \wedge \mathbf{B_i}\ \phi &\Rightarrow \mathbf{B_j}\ \phi \\ Ext_{\psi}(i,j) \wedge \mathbf{B_i}\ \phi &\Rightarrow \mathbf{B_j}\ \phi \\ Ext_{c}(i,j) \wedge \mathbf{B_i}\ \phi &\Rightarrow \mathbf{B_j}\ \phi \end{aligned}$$

où nous avons utilisé un symbole de méta-niveau pour désigner $\psi \in \{g, a, r, p\}$ et ϕ est un prédicat dans le langage interne des agents. Selon nos définitions D8.1-D8.6, il est presque immédiat de conclure que l'ordre des deux premiers termes de ces prédicats peut être inversé, car ils sont définis en utilisant des équivalences logiques. Par conséquent, pour n'importe quel théorème de la forme précédente, nous aurions pu aussi prouver les théorèmes suivants :

$$\begin{aligned} Comp_{\psi}(i,j,k) \wedge \mathbf{B_j}\ \phi &\Rightarrow \mathbf{B_i}\ \phi \\ Ext_{\psi}(i,j) \wedge \mathbf{B_j}\ \phi &\Rightarrow \mathbf{B_i}\ \phi \\ Ext_{c}(i,j) \wedge \mathbf{B_j}\ \phi &\Rightarrow \mathbf{B_i}\ \phi \end{aligned}$$

Par concision, nous ne le ferons pas. Chaque fois que nous aurons besoin d'un théorème dans cette dernière forme, nous citerons celui de la forme précédente, en disant que nous adoptons *l'hypothèse de commutativité du langage externe.*

A.4 Notions de base

Théorème A.4.1. $Comp_g(i,j,k) \wedge \mathbf{B_i}\ is_g(k,g) \Rightarrow \mathbf{B_j}\ is_g(k,g)$

Preuve :

1. $\mathbf{B_i}\ is_g(k,g)$	$\wedge$E H
2. $Comp_g(i,j,k)$	$\wedge$E H
3. $\forall g\ (\mathbf{B_i}\ is_g(k,g) \Leftrightarrow \mathbf{B_j}\ is_g(k,g))$	MP 2 D8.1
4. $\mathbf{B_i}\ is_g(k,g) \Leftrightarrow \mathbf{B_j}\ is_g(k,g)$	$\forall$E 3
5. $\mathbf{B_j}\ is_g(k,g)$	MP 1 4

□

Corollaire A.4.1. $Comp_g(i,j,k) \wedge \mathbf{B_i}\ \neg is_g(k,g) \Rightarrow \mathbf{B_j}\ \neg is_g(k,g)$

Preuve :

Supposons par absurde que $\neg\mathbf{B_j}\ \neg is_g(k,g)$. Etant donné que le langage interne est complet, nous pouvons inférer par MP en utilisant DA.1 $\mathbf{B_j}\ is_g(k,g)$. Si nous utilisons l'hypothèse de commutativité du langage externe appliquée au théorème A.4.1, nous aurons $Comp_g(i,j,k) \wedge \mathbf{B_j}\ is_g(k,g) \Rightarrow \mathbf{B_i}\ is_g(k,g)$. Comme par hypothèse $Comp_g(i,j,k)$ est vrai, nous pourrions déduire par MP $\mathbf{B_i}\ is_g(k,g)$. Ce résultat est clairement en contradiction avec l'hypothèse $\mathbf{B_i}\ \neg is_g(k,g)$, car nous supposons que le langage interne est correct. Par conséquent, nous devons avoir $\mathbf{B_j}\ \neg is_g(k,g)$, puisque le langage interne est complet.

□

Les résultats précédents peuvent être facilement étendus, de manière similaire, aux actions, aux ressources et aux plans, si nous prenons respectivement les définitions D8.2, D8.3 et D8.4 :

Théorème A.4.2. $Comp_a(i,j,k) \wedge \mathbf{B_i}\ is_a(k,a) \Rightarrow \mathbf{B_j}\ is_a(k,a)$

Corollaire A.4.2. $Comp_a(i,j,k) \wedge \mathbf{B_i}\ \neg is_a(k,a) \Rightarrow \mathbf{B_j}\ \neg is_a(k,a)$

Théorème A.4.3. $Comp_r(i,j,k) \wedge \mathbf{B_i}\ is_r(k,r) \Rightarrow \mathbf{B_j}\ is_r(k,r)$

Corollaire A.4.3. $Comp_r(i,j,k) \wedge \mathbf{B_i}\ \neg is_r(k,r) \Rightarrow \mathbf{B_j}\ \neg is_r(k,r)$

Théorème A.4.4. $Comp_p(i,j,k) \wedge \mathbf{B_i}\ is_p(k,p) \Rightarrow \mathbf{B_j}\ is_p(k,p)$

Corollaire A.4.4. $Comp_p(i,j,k) \wedge \mathbf{B_i}\ \neg is_p(k,p) \Rightarrow \mathbf{B_j}\ \neg is_p(k,p)$

En particulier, nous sommes intéressés à appliquer ces résultats aux cas où l'hypothèse de compatibilité des descriptions externes est valable. Autrement dit, nous voulons analyser les cas où k est affecté soit à i soit à j. Pour ne pas alourdir la notation, dans tous les théorèmes et corollaires qui suivent, nous considérons k comme étant variable à affecter soit à i soit à j : $(k \in \{i,j\})$.

Théorème A.4.5. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_g(k,g) \Rightarrow \mathbf{B_j}\ is_g(k,g)$

Preuve :

Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ is_g(i,g)$ — ∧E H
2. $Ext_c(i,j)$ — ∧E H
3. $Comp(i,j,i) \wedge Comp(i,j,j)$ — MP 2 D8.6
4. $Comp(i,j,i)$ — ∧E 3

5. $Comp_g(i,j,i) \wedge Comp_a(i,j,i) \wedge Comp_r(i,j,i) \wedge Comp_p(i,j,i)$ — MP 4 D8.5 $(k \mid i)$
6. $Comp_g(i,j,i)$ — $\wedge$E 5
7. $Comp_g(i,j,i) \wedge \mathbf{B_i}\ is_g(i,g)$ — $\wedge$I 6 1
8. $\mathbf{B_j}\ is_g(i,g)$ — MP 7 TA.4.1 $(k \mid i)$

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence du prédicat is_g, prendre à l'étape 4 le deuxième terme du pas 3 à la place du premier, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans les étapes 5 et 8.

□

Corollaire A.4.5. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg is_g(k,g) \Rightarrow \mathbf{B_j}\ \neg is_g(k,g)$

Preuve :
La preuve est similaire à celle du corollaire A.4.1, si nous appliquons l'hypothèse de commutativité du langage externe au théorème précédent.

□

De façon tout à fait similaire, l'extension de ces résultats aux actions, aux ressources et aux plans est triviale, si nous prenons à l'étape 4 respectivement le deuxième, troisième et quatrième terme de l'étape 3 et si nous utilisons respectivement les théorèmes A.4.2, A.4.3, et A.4.4 à l'étape 7 :

Théorème A.4.6. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_a(k,a) \Rightarrow \mathbf{B_j}\ is_a(k,a)$

Corollaire A.4.6. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg is_a(k,a) \Rightarrow \mathbf{B_j}\ \neg is_a(k,a)$

Théorème A.4.7. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_r(k,r) \Rightarrow \mathbf{B_j}\ is_r(k,r)$

Corollaire A.4.7. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg is_r(k,r) \Rightarrow \mathbf{B_j}\ \neg is_r(k,r)$

Théorème A.4.8. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_p(k,p) \Rightarrow \mathbf{B_j}\ is_p(k,p)$

Corollaire A.4.8. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg is_p(k,p) \Rightarrow \mathbf{B_j}\ \neg is_p(k,p)$

A.5 Notions auxiliaires

Théorème A.5.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ is\_plan_g(k,g,p) \Rightarrow \mathbf{B_j}\ is\_plan_g(k,g,p)$

Preuve :
Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ is\_plan_g(i,g,p)$ — $\wedge$E H

2. Dans le langage interne de l'agent i :

2.1. $is\_plan_g(i,g,p)$	1
2.2. $is_p(i,p) \wedge achieves_g(p,g)$	MP 2.1 D5.19
2.3. $is_p(i,p)$	$\wedge$E 2.2
2.4. $achieves_g(p,g)$	$\wedge$E 2.2

3. $Ext_c(i,j)$	$\wedge$E H
4. $\mathbf{B_i}\ is_p(i,p)$	2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_p(i,p)$	$\wedge$I 3 4
6. $\mathbf{B_j}\ is_p(i,p)$	MP 5 TA.4.8 $(k \mid i)$
7. $\mathbf{B_i}\ achieves_g(p,g)$	2.4
8. $\mathbf{B_j}\ achieves_g(p,g)$	MP 7 DA.2

9. Dans le langage interne de l'agent j :

9.1. $is_p(i,p)$	6
9.2. $achieves_g(p,g)$	8
9.3. $is_p(i,p) \wedge achieves_g(p,g)$	$\wedge$I 9.1 9.2
9.4. $is\_plan_g(i,g,p)$	MP 9.3 D5.19

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats is_p et is_plan_g, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ à l'étape 6. Nous devons aussi utiliser la substitution $(i \mid j)$ aux étapes 2.2 et 9.4.

□

Corollaire A.5.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ has_p(k,g) \Rightarrow \mathbf{B_j}\ has_p(k,g)$

Preuve :

Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ has_p(i,g)$	$\wedge$E H

2. Dans le langage interne de l'agent i :

2.1. $has_p(i,g)$	1
2.2. $\exists p\ is\_plan_g(i,g,p)$	MP 2.1 D5.20
2.3. $is\_plan_g(i,g,\pi)$	$\exists$E 2.2

3. $Ext_c(i,j)$ ∧E H

4. $\mathbf{B_i}\ is\_plan_g(i,g,\pi)$ 2.3

5. $Ext_c(i,j) \wedge \mathbf{B_i}\ is\_plan_g(i,g,\pi)$ ∧I 3 4

6. $\mathbf{B_j}\ is\_plan_g(i,g,\pi)$ MP 5 TA.5.1 $(k \mid i)$ $(p \mid \pi)$

7. Dans le langage interne de l'agent j :

   7.1. $is\_plan_g(i,g,\pi)$ 6

   7.2. $\exists p\ is\_plan_g(i,g,p)$ ∃I 7.1

   7.3. $has_p(i,g)$ MP 7.2 D5.20

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats has_p et is_plan_g, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans le pas 6. Nous devons aussi utiliser la substitution $(i \mid j)$ dans les étapes 2.2 et 7.3.

□

Théorème A.5.2. $Ext_c(i,j) \wedge \mathbf{B_i}\ has\_all_a(k,p) \Rightarrow \mathbf{B_j}\ has\_all_a(k,p)$

Preuve :
Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ has\_all_a(i,p)$ ∧E H

2. Dans le langage interne de l'agent i :

   2.1. $has\_all_a(i,p)$ 1

   2.2. $\forall a\ (uses_a(p,a) \Rightarrow is_a(i,a))$ MP 2.1 D5.22

   2.3. Supposons que p soit un plan qui utilise l'ensemble d'actions $\Lambda = \{\alpha_1, \ldots, \alpha_n\}$. Pour chaque action $\alpha_q \in \Lambda$ nous avons :

      2.3.1. $uses_a(p,\alpha_q)$ def

      2.3.2. $is_a(i,\alpha_q)$ ∀E 2.2

3. $Ext_c(i,j)$ ∧E H

4. Pour chaque action $\alpha_q \in \Lambda$, nous avons :

   4.1. $\mathbf{B_i}\ uses_a(p,\alpha_q)$ 2.3.1

   4.2. $\mathbf{B_j}\ uses_a(p,\alpha_q)$ MP 4.1 DA.3 $(a \mid \alpha_q)$

   4.3. $\mathbf{B_i}\ is_a(i,\alpha_q)$ 2.3.2

   4.4. $Ext_c(i,j) \wedge \mathbf{B_j}\ is_a(i,\alpha_q)$ ∧I 3 4.3

4.5. $\mathbf{B_j}\ is_a(i, \alpha_q)$ MP 4.4 TA.4.6 $(k \mid i)$ $(a \mid \alpha_q)$

5. Dans le langage interne de l'agent j :

   5.1. Pour chaque action $\alpha_q \in \Lambda$, nous avons :

      5.1.1. $uses_a(p, \alpha_q)$ 4.2

      5.1.2. $is_a(i, \alpha_q)$ 4.5

      5.1.3. $uses_a(p, \alpha_q) \Rightarrow is_a(i, \alpha_q)$ $\Rightarrow$I 5.1.1 5.1.2

   5.2. Pour toute action a, nous avons soit $a \in \Lambda$ soit $a \notin \Lambda$. Par définition, si $a \notin \Lambda$, $uses_a(p, a)$ est faux, nous pouvons déduire $uses_a(p, a) \Rightarrow is_a(i, a)$. Si, par contre, $a \in \Lambda$, nous pouvons déduire le même résultat, comme ceci a été montré dans 5.1.3. Par conséquent, nous pouvons introduire le quantificateur universel :

      $\forall a\ (uses_a(p, a) \Rightarrow is_a(i, a))$ $\forall$I 5.1.3

   5.3. $has\_all_a(i, p)$ MP 5.2 D5.22

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats is_a et has_all_a, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ à l'étape 4.5. Nous devons aussi utiliser la substitution $(i \mid j)$ aux étapes 2.2 et 5.3.

□

Théorème A.5.3. $Ext_c(i, j) \wedge \mathbf{B_i}\ needs_a(k, p, a) \Rightarrow \mathbf{B_j}\ needs_a(k, p, a)$

Preuve :

Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ needs_a(i, p, a)$ $\wedge$E H

2. Dans le langage interne de l'agent i :

   2.1. $needs_a(i, p, a)$ 1

   2.2. $uses_a(p, a) \wedge \neg is_a(i, a)$ MP 2.1 D5.21

   2.3. $uses_a(p, a)$ $\wedge$E 2.2

   2.4. $\neg is_a(i, a)$ $\wedge$E 2.2

3. $Ext_c(i, j)$ $\wedge$E H

4. $\mathbf{B_i}\ uses_a(p, a)$ 2.3

5. $\mathbf{B_j}\ uses_a(p, a)$ MP 4 DA.3

6. $\mathbf{B_i}\ \neg is_a(i, a)$ 2.4

7. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg is_a(i,a)$ $\wedge$I 3 6

8. $\mathbf{B_j}\ \neg is_a(i,a)$ MP 7 CA.4.6 $(k \mid i)$

9. Dans le langage interne de l'agent j :

   9.1. $uses_a(p,a)$ 5

   9.2. $\neg is_a(i,a)$ 8

   9.3. $uses_a(p,a) \wedge v\neg is_a(i,a)$ $\wedge$I 9.1 9.2

   9.4. $needs_a(i,p,a)$ MP 9.3 D5.21

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats is_a et $needs_a$, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans le pas 8. Nous devons aussi utiliser la substitution $(i \mid j)$ dans les étapes 2.2 et 9.4.

□

A.6 Notions sur les plans

Théorème A.6.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ aut\_plan_a(i,g,p,k) \Rightarrow \mathbf{B_j}\ aut\_plan_a(i,g,p,k)$

Preuve :
Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ aut\_plan_a(i,g,p,i)$ $\wedge$E H

2. Dans le langage interne de l'agent i :

   2.1. $aut\_plan_a(i,g,p,i)$ 1

   2.2. $is\_plan_g(i,g,p) \wedge has\_all_a(i,p)$ MP 2.1 D5.25 $(k \mid i)$

   2.3. $is\_plan_g(i,g,p)$ $\wedge$E 2.2

   2.4. $has\_all_a(i,p)$ $\wedge$E 2.2

3. $Ext_c(i,j)$ $\wedge$E H

4. $\mathbf{B_i}\ is\_plan_g(i,g,p)$ 2.3

5. $Ext_c(i,j) \wedge \mathbf{B_i}\ is\_plan_g(i,g,p)$ $\wedge$I 3 4

6. $\mathbf{B_j}\ is\_plan_g(i,g,p)$ MP 5 TA.5.1 $(k \mid i)$

7. $\mathbf{B_i}\ has\_all_a(i,p)$ 2.4

8. $Ext_c(i,j) \wedge \mathbf{B_i}\ has\_all_a(i,p)$ $\wedge$I 3 7

9. $\mathbf{B_j}\ has\_all_a(i,p)$ MP 8 TA.5.2 $(k \mid i)$

10. Dans le langage interne de l'agent i :

    10.1. $is\_plan_g(i,g,p)$ 6

    10.2. $has\_all_a(i,p)$ 9

    10.3. $is\_plan_g(i,g,p) \wedge has\_all_a(i,p)$ $\wedge$I 10.1 10.2

    10.4. $aut\_plan_a(i,g,p,i)$ MP 10.3 D5.25 $(k \mid i)$

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence du prédicat is_plan_g et dans le dernier terme de chaque occurrence du prédicat aut_plan_a, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans les étapes 2.2, 6 et 10.4.

□

Corollaire A.6.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ aut\_plan_a(j,g,p,k) \Rightarrow \mathbf{B_j}\ aut\_plan_a(j,g,p,k)$

Preuve :

La preuve est similaire à celle du théorème précédent. Nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats aut_plan_a et has_all_a, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ à l'étape 9. Nous devons aussi utiliser la substitution $(i \mid j)$ aux étapes 2.2 et 10.4.

□

Théorème A.6.2.

$Ext_c(i,j) \wedge \mathbf{B_i}\ \neg\exists p\ aut\_plan_a(i,g,p,k) \Rightarrow \mathbf{B_j}\ \neg\exists p\ aut\_plan_a(i,g,p,k)$

Preuve :

Supposons par absurde que $\neg\mathbf{B_j}\ \neg\exists p\ aut\_plan_a(i,g,p,k)$ est vrai. Etant donné que le langage interne est complet, nous pouvons inférer par MP en utilisant DA.1 $\mathbf{B_j}\ \exists p\ aut\_plan_a(i,g,p,k)$. Appelons π ce plan. Si nous utilisons l'hypothèse de commutativité du langage externe appliquée au théorème A.6.1 avec la substitution $(p \mid \pi)$, nous aurons $Ext_c(i,j) \wedge \mathbf{B_j}\ aut\_plan_a(i,g,\pi,k) \Rightarrow \mathbf{B_i}\ aut\_plan_a(i,g,\pi,k)$. Comme par hypothèse $Comp_g(i,j,k)$ est vrai, nous pourrions déduire par MP $\mathbf{B_i}\ aut\_plan_a(i,g,\pi,k)$. Ce résultat est clairement en contradiction avec l'hypothèse $\mathbf{B_i}\ \neg\exists p\ aut\_plan_a(i,g,p,k)$, car nous supposons que le langage interne est correct. Par conséquent, nous devons avoir $\mathbf{B_j}\ \neg\exists p\ aut\_plan_a(i,g,p,k)$, puisque le langage interne est complet.

□

Corollaire A.6.2.

$Ext_c(i,j) \wedge \mathbf{B_i}\ \neg\exists p\ aut\_plan_a(j,g,p,k) \Rightarrow \mathbf{B_j}\ \neg\exists p\ aut\_plan_a(j,g,p,k)$

Preuve :
La preuve est similaire à celle du théorème précédent, si nous utilisons le corollaire A.6.1 à la place du théorème A.6.1.

□

Théorème A.6.3.
$Ext_c(i,j) \wedge \mathbf{B_i}\ basic\_dep_a(i,j,g,p,a) \Rightarrow \mathbf{B_j}\ basic\_dep_a(i,j,g,p,a)$

Preuve :

1. $\mathbf{B_i}\ basic\_dep_a(i,j,g,p,a)$ — ∧E H
2. Dans le langage interne de l'agent i :
   - 2.1. $basic\_dep_a(i,j,g,p,a)$ — 1
   - 2.2. $achieves_g(p,g) \wedge needs_a(i,p,a) \wedge is_a(j,a)$ — MP 2.1 D5.29
   - 2.3. $achieves_g(p,g)$ — ∧E 2.2
   - 2.4. $needs_a(i,p,a)$ — ∧E 2.2
   - 2.5. $is_a(j,a)$ — ∧E 2.2
3. $\mathbf{B_i}\ achieves_g(p,g)$ — 2.3
4. $\mathbf{B_j}\ achieves_g(p,g)$ — MP 3 DA.2
5. $Ext_c(i,j)$ — ∧E H
6. $\mathbf{B_i}\ needs_a(i,p,a)$ — 2.4
7. $Ext_c(i,j) \wedge \mathbf{B_i}\ needs_a(i,p,a)$ — ∧I 5 6
8. $\mathbf{B_j}\ needs_a(i,p,a)$ — MP 7 TA.5.3 $(k \mid i)$
9. $\mathbf{B_i}\ is_a(j,a)$ — 2.5
10. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_a(j,a)$ — ∧I 5 9
11. $\mathbf{B_j}\ is_a(j,a)$ — MP 10 TA.4.6 $(k \mid j)$
12. Dans le langage interne de l'agent j :
    - 12.1. $achieves_g(p,g)$ — 4
    - 12.2. $needs_a(i,p,a)$ — 8
    - 12.3. $is_a(j,a)$ — 11
    - 12.4. $achieves_g(p,g) \wedge needs_a(i,p,a) \wedge is_a(j,a)$ — ∧I 12.1 12.2 12.3
    - 12.5. $basic\_dep_a(i,j,g,p,a)$ — MP 12.4 D5.29

□

Corollaire A.6.3.
$Ext_c(i,j) \wedge \mathbf{B_i}\ basic\_dep_a(j,i,g,p,a) \Rightarrow \mathbf{B_j}\ basic\_dep_a(j,i,g,p,a)$

Preuve :
La preuve est similaire à celle du théorème précédent. Nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats $basic\_dep_a$ et $needs_a$, j par i dans le premier terme de chaque occurrence du prédicat is_a, et faire les substitutions $(k \mid j)$ à la place de $(k \mid i)$ à l'étape 8 et $(k \mid i)$ à la place de $(k \mid j)$ à l'étape 11. Nous devons aussi utiliser les substitutions $(i \mid j)$ et $(j \mid i)$ aux étapes 2.2 et 12.5.

□

Théorème A.6.4.
$Ext_c(i,j) \wedge \mathbf{B_i}\ dep\_plan_a(i,j,g,p,k) \Rightarrow \mathbf{B_j}\ dep\_plan_a(i,j,g,p,k)$

Preuve :
Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ dep\_plan_a(i,j,g,p,i)$ — ∧E H
2. Dans le langage interne de l'agent i :
   - 2.1. $dep\_plan_a(i,j,g,p,i)$ — 1
   - 2.2. $is_p(i,p) \wedge \exists a\ basic\_dep_a(i,j,g,p,a)$ — MP 2.1 D5.27 $(k \mid i)$
   - 2.3. $is_p(i,p)$ — ∧E 2.2
   - 2.4. $\exists a\ basic\_dep_a(i,j,g,p,a)$ — ∧E 2.2
   - 2.5. $basic\_dep_a(i,j,g,p,\alpha)$ — ∃E 2.4
3. $Ext_c(i,j)$ — ∧E H
4. $\mathbf{B_i}\ is_p(i,p)$ — 2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_p(i,p)$ — ∧I 3 4
6. $\mathbf{B_j}\ is_p(i,p)$ — MP 5 TA.4.8 $(k \mid i)$
7. $basic\_dep_a(i,j,g,p,\alpha)$ — 2.5
8. $Ext_c(i,j) \wedge \mathbf{B_i}\ basic\_dep_a(i,j,g,p,\alpha)$ — ∧I 3 7
9. $\mathbf{B_j}\ basic\_dep_a(i,j,g,p,\alpha)$ — MP 8 TA.6.3 $(a \mid \alpha)$
10. Dans le langage interne de l'agent j :

10.1. $is_p(i,p)$ 6

10.2. $basic\_dep_a(i,j,g,p,\alpha)$ 9

10.3. $\exists a\ basic\_dep_a(i,j,g,p,a)$ $\exists$I 10.2

10.4. $is_p(i,p) \wedge \exists a\ basic\_dep_a(i,j,g,p,a)$ $\wedge$I 10.1 10.3

10.5. $dep\_plan_a(i,j,g,p,i)$ MP 10.4 D5.27 $(k \mid i)$

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence du prédicat is_p et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans les étapes 2.2, 6 et 10.5.

□

Corollaire A.6.4.
$Ext_c(i,j) \wedge \mathbf{B_i}\ dep\_plan_a(j,i,g,p,k) \Rightarrow \mathbf{B_j}\ dep\_plan_a(j,i,g,p,k)$

Preuve :
La preuve est similaire à celle du théorème précédent. Nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats dep_plan_a et $basic\_dep_a$, j par i dans le deuxième terme de chaque occurrence de ces mêmes prédicats. À l'étape 9, on utilise le corollaire A.6.3 à la place du théorème A.6.3. Nous devons aussi utiliser les substitutions $(i \mid j)$ et $(j \mid i)$ aux étapes 2.2 et 10.5.

□

A.7 Notions d'autonomie

Théorème A.7.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ aut_a(i,g,k) \Rightarrow \mathbf{B_j}\ aut_a(i,g,k)$

Preuve :
Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ aut_a(i,g,i)$ $\wedge$E H

2. Dans le langage interne de l'agent i :

   2.1. $aut_a(i,g,i)$ 1

   2.2. $is_g(i,g) \wedge has_p(i,g) \wedge \exists p\ aut\_plan_a(i,g,p,i)$ MP 2.1 D5.31 $(k \mid i)$

   2.3. $is_g(i,g)$ $\wedge$E 2.2

   2.4. $has_p(i,g)$ $\wedge$E 2.2

   2.5. $\exists p\ aut\_plan_a(i,g,p,i)$ $\wedge$E 2.2

   2.6. $aut\_plan_a(i,g,\pi,i)$ $\exists$E 2.5

3. $Ext_c(i,j)$ $\wedge$E H

4. $\mathbf{B_i}\ is_g(i,g)$ — 2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_g(i,g)$ — $\wedge$I 3 4
6. $\mathbf{B_j}\ is_g(i,g)$ — MP 5 TA.4.5 $(k \mid i)$
7. $\mathbf{B_i}\ has_p(i,g)$ — 2.4
8. $Ext_c(i,j) \wedge \mathbf{B_i}\ has_p(i,g)$ — $\wedge$I 3 7
9. $\mathbf{B_j}\ has_p(i,g)$ — MP 8 CA.5.1 $(k \mid i)$
10. $\mathbf{B_i}\ aut\_plan_a(i,g,\pi,i)$ — 2.6
11. $Ext_c(i,j) \wedge \mathbf{B_i}\ aut\_plan_a(i,g,\pi,i)$ — $\wedge$I 3 10
12. $\mathbf{B_j}\ aut\_plan_a(i,g,\pi,i)$ — MP 11 TA.6.1 $(k \mid i)$ $(p \mid \pi)$
13. Dans le langage interne de l'agent j :
    - 13.1. $aut\_plan_a(i,g,\pi,i)$ — 12
    - 13.2. $\exists p\ aut\_plan_a(i,g,p,i)$ — $\exists$I 13.1
    - 13.3. $is_g(i,g)$ — 6
    - 13.4. $has_p(i,g)$ — 9
    - 13.5. $is_g(i,g) \wedge has_p(i,g) \wedge \exists p\ aut\_plan_a(i,g,p,i)$ — $\wedge$I 13.3 13.4 13.2
    - 13.6. $aut_a(i,g,i)$ — MP 13.5 D5.31 $(k \mid i)$

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence du prédicat has_p et dans le dernier terme de chaque occurrence des prédicats aut_a et aut_plan_a, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans les pas 2.2, 9, 12 et 13.6.

□

Corollaire A.7.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ aut_a(j,g,k) \Rightarrow \mathbf{B_j}\ aut_a(j,g,k)$

Preuve :

La preuve est similaire à celle du théorème précédent. Nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats aut_a, aut_plan_a et is_g, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ à l'étape 6. À l'étape 12, on utilise le corollaire A.6.1 à la place du théorème A.6.1. Nous devons aussi utiliser la substitution $(i \mid j)$ aux étapes 2.2 et 13.6.

□

A.8 Relations de dépendance

Théorème A.8.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ dep_a(i,g,k) \Rightarrow \mathbf{B_j}\ dep_a(i,g,k)$

Preuve :
Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ dep_a(i,g,i)$ — ∧E H
2. Dans le langage interne de l'agent i :
   - 2.1. $dep_a(i,g,i)$ — 1
   - 2.2. $is_g(i,g) \wedge has_p(i,g) \wedge \neg\exists p\ aut\_plan_a(i,g,p,i)$ — MP 2.1 D5.34 $(k \mid i)$
   - 2.3. $is_g(i,g)$ — ∧E 2.2
   - 2.4. $has_p(i,g)$ — ∧E 2.2
   - 2.5. $\neg\exists p\ aut\_plan_a(i,g,p,i)$ — ∧E 2.2
3. $Ext_c(i,j)$ — ∧E H
4. $\mathbf{B_i}\ is_g(i,g)$ — 2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\ is_g(i,g)$ — ∧I 3 4
6. $\mathbf{B_j}\ is_g(i,g)$ — MP 5 TA.4.5 $(k \mid i)$
7. $\mathbf{B_i}\ has_p(i,g)$ — 2.4
8. $Ext_c(i,j) \wedge \mathbf{B_i}\ has_p(i,g)$ — ∧I 3 7
9. $\mathbf{B_j}\ has_p(i,g)$ — MP 8 CA.5.1 $(k \mid i)$
10. $\mathbf{B_i}\ \neg\exists p\ aut\_plan_a(i,g,p,i)$ — 2.5
11. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg\exists p\ aut\_plan_a(i,g,p,i)$ — ∧I 3 10
12. $\mathbf{B_j}\ \neg\exists p\ aut\_plan_a(i,g,p,i)$ — MP 11 TA.6.2 $(k \mid i)$
13. Dans le langage interne de l'agent j :
    - 13.1. $is_g(i,g)$ — 6
    - 13.2. $has_p(i,g)$ — 9
    - 13.3. $\neg\exists p\ aut\_plan_a(i,g,p,i)$ — 12
    - 13.4. $is_g(i,g) \wedge has_p(i,g) \wedge \neg\exists p\ aut\_plan_a(i,g,p,i)$ — ∧I 13.1 13.2 13.3
    - 13.5. $dep_a(i,g,i)$ — MP 13.4 D5.34 $(k \mid i)$

Pour $k = j$, nous devons remplacer i par j dans le premier terme de chaque occurrence du prédicat has_p et dans le dernier terme de chaque occurrence des prédicats dep_a et aut_plan_a, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans les pas 2.2, 9, 12 et 13.5.

□

Corollaire A.8.1. $Ext_c(i,j) \wedge \mathbf{B_i}\, dep_a(j,g,k) \Rightarrow \mathbf{B_j}\, dep_a(j,g,k)$

Preuve :

La preuve est similaire à celle du théorème précédent. Nous devons remplacer i par j dans le premier terme de chaque occurrence des prédicats dep_a, aut_plan_a et is_g, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ à l'étape 6. À l'étape 12, on utilise le corollaire A.6.2 à la place du théorème A.6.2. Nous devons aussi utiliser la substitution $(i \mid j)$ aux étapes 2.2 et 13.5.

□

Théorème A.8.2. $Ext_c(i,j) \wedge \mathbf{B_i}\, dep\_on_a(i,j,g,k) \Rightarrow \mathbf{B_j}\, dep\_on_a(i,j,g,k)$

Preuve :

Pour $k = i$, nous avons :

1. $\mathbf{B_i}\, dep\_on_a(i,j,g,i)$ — ∧E H
2. Dans le langage interne de l'agent i :
   - 2.1. $dep\_on_a(i,j,g,i)$ — 1
   - 2.2. $dep_a(i,g,i) \wedge \exists p\, dep\_plan_a(i,j,g,p,i)$ — MP 2.1 D5.37 $(k \mid i)$
   - 2.3. $dep_a(i,g,i)$ — ∧E 2.2
   - 2.4. $\exists p\, dep\_plan_a(i,j,g,p,i)$ — ∧E 2.2
   - 2.5. $dep\_plan_a(i,j,g,\pi,i)$ — ∃E 2.4
3. $Ext_c(i,j)$ — ∧E H
4. $\mathbf{B_i}\, dep_a(i,g,i)$ — 2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\, dep_a(i,g,i)$ — ∧I 3 4
6. $\mathbf{B_j}\, dep_a(i,g,i)$ — MP 5 TA.8.1 $(k \mid i)$
7. $\mathbf{B_i}\, dep\_plan_a(i,j,g,\pi,i)$ — 2.5
8. $Ext_c(i,j) \wedge \mathbf{B_i}\, dep\_plan_a(i,j,g,\pi,i)$ — ∧I 3 7
9. $\mathbf{B_j}\, dep\_plan_a(i,j,g,\pi,i)$ — MP 8 TA.6.4 $(k \mid i)$ $(p \mid \pi)$
10. Dans le langage interne de l'agent j :

10.1. $dep\_plan_a(i,j,g,\pi,i)$ 9

10.2. $\exists p\ dep\_plan_a(i,j,g,p,i)$ $\exists$I 10.1

10.3. $dep_a(i,g,i)$ 6

10.4. $dep_a(i,g,i) \wedge \exists p\ dep\_plan_a(i,j,g,p,i)$ $\wedge$I 10.3 10.2

10.5. $dep\_plan_a(i,j,g,p,i)$ MP 10.4 D5.37 $(k \mid i)$

Pour $k = j$, nous devons remplacer i par j dans le dernier terme de chaque occurrence des prédicats dep_plan_a, dep_on_a et dep_a, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans les étapes 2.2, 6, 9 et 10.5.

□

Corollaire A.8.2. $Ext_c(i,j) \wedge \mathbf{B_i}\ dep\_on_a(j,i,g,k) \Rightarrow \mathbf{B_j}\ dep\_on_a(j,i,g,k)$

Preuve :

La preuve est similaire à celle du théorème précédent. Nous devons remplacer i par j et j par i respectivement dans le premier et le deuxième termes de chaque occurrence des prédicats dep_a, dep_plan_a et dep_on_a. Dans les étapes 6 et 9, on utilise respectivement les corollaires A.8.1 et A.6.4 à la place des théorèmes A.8.1 et A.6.4. Nous devons aussi utiliser les substitutions $(i \mid j)$ et $(j \mid i)$ dans les étapes 2.2 et 10.5.

□

Théorème A.8.3. $Ext_c(i,j) \wedge \mathbf{B_i} \neg dep\_on_a(i,j,g,k) \Rightarrow \mathbf{B_j} \neg dep\_on_a(i,j,g,k)$

Preuve :

La preuve est similaire à celle du corollaire A.4.1, si nous appliquons l'hypothèse de commutativité du langage externe au théorème A.8.2.

□

Corollaire A.8.3. $Ext_c(i,j) \wedge \mathbf{B_i} \neg dep\_on_a(j,i,g,k) \Rightarrow \mathbf{B_j} \neg dep\_on_a(j,i,g,k)$

Preuve :

La preuve est similaire à celle du corollaire A.4.1, si nous appliquons l'hypothèse de commutativité du langage externe au corollaire A.8.2.

□

A.9 Dépendance mutuelle et dépendance réciproque

Théorème A.9.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ MD(i,j,g,k) \Rightarrow \mathbf{B_j}\ MD(j,i,g,k)$

Preuve :

Pour $k = i$, nous avons :

1. $\mathbf{B_i}\ MD(i,j,g,i)$ $\wedge$E H
2. Dans le langage interne de l'agent i :
   - 2.1. $MD(i,j,g,i)$ 1
   - 2.2. $dep\_on_a(i,j,g,i) \wedge dep\_on_a(j,i,g,i)$ MP 2.1 D5.40 $(k \mid i)$
   - 2.3. $dep\_on_a(i,j,g,i)$ $\wedge$E 2.2
   - 2.4. $dep\_on_a(j,i,g,i)$ $\wedge$E 2.2
3. $Ext_c(i,j)$ $\wedge$E H
4. $\mathbf{B_i}\ dep\_on_a(i,j,g,i)$ 2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\ dep\_on_a(i,j,g,i)$ $\wedge$I 3 4
6. $\mathbf{B_j}\ dep\_on_a(i,j,g,i)$ MP 5 TA.8.2 $(k \mid i)$
7. $\mathbf{B_i}\ dep\_on_a(j,i,g,i)$ 2.4
8. $Ext_c(i,j) \wedge \mathbf{B_i}\ dep\_on_a(j,i,g,i)$ $\wedge$I 3 7
9. $\mathbf{B_j}\ dep\_on_a(j,i,g,i)$ MP 8 CA.8.2 $(k \mid i)$
10. Dans le langage interne de l'agent j :
    - 10.1. $dep\_on_a(j,i,g,i)$ 9
    - 10.2. $dep\_on_a(i,j,g,i)$ 6
    - 10.3. $dep\_on_a(j,i,g,i) \wedge dep\_on_a(i,j,g,i)$ $\wedge$I 10.1 10.2
    - 10.4. $MD(j,i,g,i)$ MP 10.3 D5.40 $(i \mid j)$ $(j \mid i)$ $(k \mid i)$

Pour $k = j$, nous devons remplacer i par j dans le dernier terme de chaque occurrence des prédicats MD et dep_on_a, et faire la substitution $(k \mid j)$ à la place de $(k \mid i)$ dans les étapes 2.2, 6, 9 et 10.4.

□

Corollaire A.9.1. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg MD(i,j,g,k) \Rightarrow \mathbf{B_j}\ \neg MD(j,i,g,k)$

Preuve :

La preuve est similaire à celle du corollaire A.4.1, si nous appliquons l'hypothèse de commutativité du langage externe au théorème A.9.1.

□

Théorème A.9.2. $Ext_c(i,j) \wedge \mathbf{B_i}\ RD(i,j,g,g',k) \Rightarrow \mathbf{B_j}\ RD(j,i,g',g,k)$

Preuve :

La preuve est similaire à celle du théorème précédent, la différence étant que les deux prédicats dep_on_a ne sont pas apliqués au même but. Nous devons utiliser la définition D5.41 à la place de la définition D5.40 dans les étapes 2.2 et 10.4. Nous avons besoin aussi de prouver dans le langage externe que les deux agents sont capables de faire la distinction entre les buts g et g', ce qui est accompli par l'utilisation de la définition DA.5.

□

Corollaire A.9.2. $Ext_c(i,j) \wedge \mathbf{B_i}\, \neg RD(i,j,g,g',k) \Rightarrow \mathbf{B_j}\, \neg RD(j,i,g',g,k)$

Preuve :

La preuve est similaire à celle du corollaire A.4.1, si nous appliquons l'hypothèse de commutativité du langage externe au théorème A.9.2.

□

A.10 Situations de dépendance

Théorème A.10.1. $Ext_c(i,j) \wedge \mathbf{B_i}\, MBMD(i,j,g) \Rightarrow \mathbf{B_j}\, MBMD(j,i,g)$

Preuve :

1. $\mathbf{B_i}\, MBMD(i,j,g)$ — ∧E H
2. Dans le langage interne de l'agent i :
   - 2.1. $MBMD(i,j,g)$ — 1
   - 2.2. $MD(i,j,g,i) \wedge MD(i,j,g,j)$ — MP 2.1 D5.48
   - 2.3. $MD(i,j,g,i)$ — ∧E 2.2
   - 2.4. $MD(i,j,g,j)$ — ∧E 2.2
3. $Ext_c(i,j)$ — ∧E H
4. $\mathbf{B_i}\, MD(i,j,g,i)$ — 2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\, MD(i,j,g,i)$ — ∧I 3 4
6. $\mathbf{B_j}\, MD(j,i,g,i)$ — MP 5 TA.9.1 $(k \mid i)$
7. $\mathbf{B_i}\, MD(i,j,g,j)$ — 2.4
8. $Ext_c(i,j) \wedge \mathbf{B_i}\, MD(i,j,g,j)$ — ∧I 3 7
9. $\mathbf{B_j}\, MD(j,i,g,j)$ — MP 8 TA.9.1 $(k \mid i)$
10. Dans le langage interne de l'agent j :

10.1. $MD(j,i,g,i)$	6
10.2. $MD(j,i,g,j)$	9
10.3. $MD(j,i,g,j) \wedge MD(j,i,g,i)$	$\wedge$I 10.2 10.1
10.4. $MBMD(j,i,g)$	MP 10.3 D5.48 $(i \mid j)$ $(j \mid i)$

□

Corollaire A.10.1. $\mathbf{B_i}\ MBMD(i,j,g) \wedge \neg \mathbf{B_j}\ MBMD(j,i,g) \Rightarrow \neg Ext_c(i,j)$

Preuve :

1. $Ext_c(i,j) \wedge \mathbf{B_i}\ MBMD(i,j,g) \Rightarrow \mathbf{B_j}\ MBMD(j,i,g)$	TA.10.1
2. $\neg(Ext_c(i,j) \wedge \mathbf{B_i}\ MBMD(i,j,g)) \vee \mathbf{B_j}\ MBMD(j,i,g)$	IM 1
3. $\neg Ext_c(i,j) \vee \neg \mathbf{B_i}\ MBMD(i,j,g) \vee \mathbf{B_j}\ MBMD(j,i,g)$	DM 2
4. $\neg \mathbf{B_i}\ MBMD(i,j,g) \vee \mathbf{B_j}\ MBMD(j,i,g) \vee \neg Ext_c(i,j)$	CM 3
5. $\neg \mathbf{B_i}\ MBMD(i,j,g) \vee \neg\neg \mathbf{B_j}\ MBMD(j,i,g) \vee \neg Ext_c(i,j)$	$\neg$I 4
6. $\neg(\mathbf{B_i}\ MBMD(i,j,g) \wedge \neg \mathbf{B_j}\ MBMD(j,i,g)) \vee \neg Ext_c(i,j)$	DM 5
7. $\mathbf{B_i}\ MBMD(i,j,g) \wedge \neg \mathbf{B_j}\ MBMD(j,i,g) \Rightarrow \neg Ext_c(i,j)$	IM 6

□

Théorème A.10.2. $Ext_c(i,j) \wedge \mathbf{B_i}\ MBRD(i,j,g,g') \Rightarrow \mathbf{B_j}\ MBRD(j,i,g',g)$

Preuve :

La preuve est presque identique à celle du théorème précédent, la différence étant que nous devons utiliser la définition D5.50 à la place de la définition D5.48 dans les étapes 2.2 et 10.4 et le théorème A.9.2 à la place du théorème A.9.1 dans les pas 6 et 9.

□

Corollaire A.10.2. $\mathbf{B_i}\ MBRD(i,j,g,g') \wedge \neg \mathbf{B_j}\ MBRD(j,i,g',g) \Rightarrow \neg Ext_c(i,j)$

Preuve :

La preuve est similaire à celle du corollaire précédent, si à l'étape 1 nous utilisons le théorème A.10.2 à la place du théorème A.10.1.

□

Théorème A.10.3. $Ext_c(i,j) \wedge \mathbf{B_i}\ LBMD(i,j,g) \Rightarrow \neg \mathbf{B_j}\ LBMD(j,i,g)$

Preuve :

1. $\mathbf{B_i}\ LBMD(i,j,g)$ — $\wedge$E H
2. Dans le langage interne de l'agent i :
   - 2.1. $LBMD(i,j,g)$ — 1
   - 2.2. $MD(i,j,g,i) \wedge \neg MD(i,j,g,j)$ — MP 2.1 D5.47
   - 2.3. $\neg MD(i,j,g,j)$ — $\wedge$E 2.2
3. $Ext_c(i,j)$ — $\wedge$E H
4. $\mathbf{B_i}\ \neg MD(i,j,g,j)$ — 2.3
5. $Ext_c(i,j) \wedge \mathbf{B_i}\ \neg MD(i,j,g,j)$ — $\wedge$I 3 4
6. $\mathbf{B_j}\ \neg MD(j,i,g,j)$ — MP 5 CA.9.1 $(k \mid j)$
7. Dans le langage interne de l'agent j :
   - 7.1. $\neg MD(j,i,g,j)$ — 6
   - 7.2. $\neg(MD(j,i,g,j) \wedge \neg MD(j,i,g,i))$ — $\wedge$I 7.1
   - 7.3. $\neg LBMD(j,i,g)$ — SU D5.47 7.2

□

Corollaire A.10.3. $\mathbf{B_i}\ LBMD(i,j,g) \wedge \mathbf{B_j}\ LBMD(j,i,g) \Rightarrow \neg Ext_c(i,j)$

Preuve :

1. $Ext_c(i,j) \wedge \mathbf{B_i}\ LBMD(i,j,g) \Rightarrow \neg \mathbf{B_j}\ LBMD(j,i,g)$ — TA.10.3
2. $\neg(Ext_c(i,j) \wedge \mathbf{B_i}\ LBMD(i,j,g)) \vee \neg \mathbf{B_j}\ LBMD(j,i,g)$ — IM 1
3. $\neg Ext_c(i,j) \vee \neg \mathbf{B_i}\ LBMD(i,j,g) \vee \neg \mathbf{B_j}\ LBMD(j,i,g)$ — DM 2
4. $\neg \mathbf{B_i}\ LBMD(i,j,g) \vee \neg \mathbf{B_j}\ LBMD(j,i,g) \vee \neg Ext_c(i,j)$ — CM 3
5. $\neg(\mathbf{B_i}\ LBMD(i,j,g) \wedge \mathbf{B_j}\ LBMD(j,i,g)) \vee \neg Ext_c(i,j)$ — DM 4
6. $\mathbf{B_i}\ LBMD(i,j,g) \wedge \mathbf{B_j}\ LBMD(j,i,g) \Rightarrow \neg Ext_c(i,j)$ — IM 5

□

Théorème A.10.4. $Ext_c(i,j) \wedge \mathbf{B_i}\ LBRD(i,j,g,g') \Rightarrow \neg \mathbf{B_j}\ LBRD(j,i,g',g)$

Preuve :

La preuve est presque identique à celle du théorème précédent, la différence étant que nous devons utiliser la définition D5.49 à la place de la définition D5.47 dans les étapes 2.2 et 7.3 et le corollaire A.9.2 à la place du corollaire A.9.1 dans le pas 6.

□

Corollaire A.10.4. $\mathbf{B_i}\ LBRD(i,j,g,g') \wedge \mathbf{B_j}\ LBRD(j,i,g',g) \Rightarrow \neg Ext_c(i,j)$

Preuve :

La preuve est similaire à celle du corollaire précédent, si à l'étape 1 nous utilisons le théorème A.10.4 à la place du théorème A.10.3.

□

Annexe B

Description de l'organisation du système EB

Dans cette annexe, nous utilisons notre modèle introduit dans le chapitre 5 pour décrire l'organisation du système EB [SJ92]. Dans la section B.1, nous présentons les caractéristiques principales de ce système. L'application de notre modèle pour le décrire est présentée dans la section B.2. Enfin, dans la section B.3 nous montrons les résultats de simulation de ce modèle en utilisant le simulateur DEPNET.

B.1 Description du système

L'objectif du système EB [SJ92], développé par l'USP (Université de São Paulo) pour la CESP (Compagnie d'Électricité de l'État de São Paulo, Brésil), est de recueillir, maintenir et produire des rapports sur des indications hydrométéorologiques de 3 grands bassins fluviaux(Pardo, Paranapanema et Tietê). Ce système a été conçu selon l'approche RDP, décrite dans la section 2.2.1.

Même si ce système n'a pas été conçu selon un approche SMA, nous pouvons bien lui imputer des notions telles que des buts et des plans. Comme présenté dans [Sho91], n'importe quel système peut être considéré comme étant intentionnel, la question est de savoir s'il est intéressant de l'analyser dans cette optique. Quand le système est parfaitement compréhensible sans avoir besoin d'utiliser ces notions, il n'est pas question de le faire. Notre but ici est d'illustrer comment un modèle de dépendances peut servir comme formalisme pour exprimer l'organisation d'une société, en prévoyant les interactions entre des divers agents du système.

Les indications hydrométéorologiques du système EB sont obtenues périodiquement par des cycles de balayage, en interrogeant un certain nombre de sites éloignés. Cette opération est accomplie en utilisant 2 réseaux de communication, dont l'un en VHF et l'autre en micro-ondes. Les sites éloignés peuvent contenir 3 types d'instruments : *pluviométriques*, *fluviométriques* et *de supervision*. Les instruments pluviométriques enregistrent le flux accumulé de pluie

depuis la dernière action de "reset", prise comme valeur de référence. Les instruments fluviométriques mesurent le niveau de l'eau dans un site particulier d'une rivière. Finalement, les instruments de supervision en registrent des informations supplémentaires sur les sites éloignés, telles que l'état des portes et lumières. Actuellement, le réseau est composé de 211 instruments.

Après chaque cycle de balayage, les données sont enregistrées sur un disque dur pour être utilisées dans un traitement statistique et envoyées à un autre système, appelé NTR, responsable de la planification des actions à moyen et long terme, du type ouverture/fermeture des vannes des certains barrages. Le système permet aussi de vider le disque dur dans des disques flexibles, quand celui-ci a atteint son limite de capacité.

Le système EB est composé de 11 processus concurrents, représentés dans la figure B.1, dont les fonctions principales sont les suivantes :

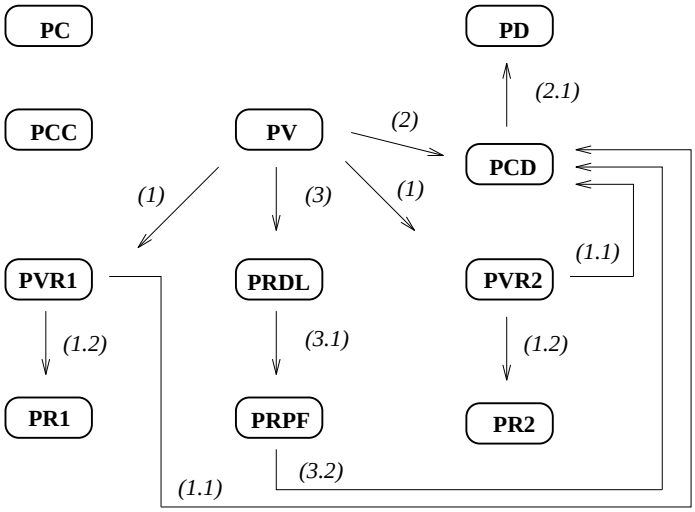


Figure B.1 : Système EB

1. *PCC* : interface homme-machine du système, qui offre à l'opérateur plusieurs actions, telles que la programmation des heures et de la périodicité des cycles de balayage (par type d'instrument et par bassin), l'habilitation/inhibition du balayage d'un instrument particulier (ce qui est utile lorsqu'un instrument fournit des indications erronées), la programmation du nombre limite de tentatives de lecture des instruments, l'interrogation manuelle de la valeur d'un instrument particulier, l'affichage d'un rapport à l'écran etc ;

2. *PCD* : base de données du système, qui garde les informations sur 3 cycles de balayage : le plus récent, ce qui a eu lieu il y a une heure et celui qui a eu lieu à l'instant de référence (la dernière action de "reset"). Les données de ces trois cycles sont maintenues car les rapports qui doivent être générés

calculent le flux accumulé (dans le cas des instruments pluviométriques) ou bien la variation de niveau de l'eau dans un site particulier (dans le cas des instruments fluviométriques) depuis la dernière heure et depuis l'instant de référence (généralement fixé à chaque 24 heures) ;

3. *PC* : communication avec le système NTR, activé quand ce dernier en fait la demande explicite ;

4. *PR1* : interfaçage avec le réseau VHF, donc responsable soit pour la génération de commandes à envoyer physiquement aux sites éloignés soit pour la décodification des réponses envoyées par ces derniers ;

5. *PR2* : interfaçage avec le réseau micro-ondes, avec les mêmes fonctions que $PR1$;

6. *PD* : gestion du disque dur et du vidage de ce dernier dans les disques flexibles ;

7. *PRDL* : gestion de l 'imprimante ;

8. *PRDF* : élaboration des rapports ;

9. *PV* : gestion des cycles de balayage, qui sont contrôlés par une horloge interne ;

10. *PVR1* : exécution des cycles de balayage dans le réseau VHF ;

11. *PVR2* : exécution des cycles de balayage dans le réseau micro-ondes.

Les arcs représentés dans la figure B.1 montrent un cycle de contrôle du système, correspondant à l'activation d'un cycle de balayage. Nous ne représentons que l'activation d'une commande, à laquelle correspond toujours une réponse que nous n'avons pas explicitement représentée dans la figure. L'agent PV active les agents $PVR1$ et $PVR2$ (1), qui doivent être exécutés de façon concurrente, pour une question de performance. Chacun de deux demandent à l'agent PCD (1.1) de leur envoyer la liste des instruments à être interrogés pendant le cycle, après quoi ils activent respectivement les agents $PR1$ et $PR2$ (1.2) pour effectuer la lecture des indications de ces instruments. Ce dernier pas est répété pour chaque instrument appartenant à la liste. Après que tous les instruments aient été interrogés, l'agent PV demande à l'agent PCD (2) d'enregistrer les informations du cycle courant, après quoi ce dernier active l'agent PD (2.1) pour que les données du dernier cycle de balayage soient stockées dans le disque dur. Finalement, l'agent PV demande à l'agent $PRDL$ (3) d'imprimer les rapports. Pour cela ce dernier demande à l'agent $PRPF$ (3.1) de les générer. Pour être capable de les générer, celui-ci demande à l'agent PCD (3.2) les données correspondantes aux 3 cycles de balayage décrites ci-dessus.

B.2 Application du modèle

En utilisant notre définition de description externe telle qu'elle a été proposée dans la section 5.1, nous pouvons représenter les divers buts, actions, ressources et plans de chaque agent de ce système, comme le montre l'exemple B.1.

Exemple B.1 : Description externe du système EB.

Agent	Buts	Actions	Ressources
PV	scan	—	—
$PVR1$	scan_vhf	scan_vhf	—
$PVR2$	scan_mv	scan_mv	—
$PR1$	get_vhf(addr_vhf)	get_vhf read_vhf	—
$PR2$	get_mv(addr_mv)	get_mv read_mv	—
PCD	update	update write_memory	addr_vhf addr_mv curr_ind
PD	update_disk	update_disk write_disk	—
$PRDL$	report	report print_report	—
$PRPF$	make_report	make_report calculate_report	—

Agents	Plans		
PV	scan()	:=	scan_vhf(), scan_mv(), update(), report().
$PVR1$	scan_vhf()	:=	get_vhf(addr_vhf)*.
$PVR2$	scan_mv()	:=	get_mv(addr_mv)*.
$PR1$	get_vhf(addr_vhf)	:=	read_vhf(addr_vhf).
$PR2$	get_mv(addr_mv)	:=	read_mv(addr_mv).
PCD	update()	:=	write_memory(curr_ind), update_disk().
PD	update_disk()	:=	write_disk(curr_ind).
$PRDL$	report()	:=	make_report(), print_report().
$PRPF$	make_report()	:=	calculate_report(curr_ind).

Dans cette table, nous avons représenté les données comme des ressources, et les diverses tâches qui doivent être accomplies par chacun des agents comme des actions. Dans le contexte de ce système, les buts correspondent aux actions qui doivent être activées à un certain moment. En ce qui concerne les plans des agents $PRV1$ et $PRV2$, un astérisque représente le fait que l'action concernée doit être répétée plusieurs fois dans un plan, car notre formalisme de base ne permet pas de représenter des itérations d'actions. Finalement, il faut remarquer que cette description externe n'existe pas au sein des agents du système, elle est implicite dans l'esprit du concepteur.

En utilisant les définitions de dépendance et autonomie introduites dans le chapitre 5, nous pouvons observer que les formules suivantes sont valables :

- en ce qui concerne l'autonomie et la dépendance vis-à-vis des actions :
  $dep_a(PV, scan, designer)$
  $dep_a(PVR1, scan\_vhf, designer)$
  $dep_a(PVR2, scan\_mv, designer)$
  $aut_a(PR1, get\_vhf, designer)$
  $aut_a(PR2, get\_mv, designer)$
  $dep_a(PCD, update, designer)$
  $aut_a(PD, update\_disk, designer)$
  $dep_a(PRDL, report, designer)$
  $aut_a(PRPF, make\_report, designer)$

- en ce qui concerne l'autonomie et la dépendance vis-à-vis des ressources :
  $aut_r(PV, scan, designer)$
  $dep_r(PVR1, scan\_vhf, designer)$
  $dep_r(PVR2, scan\_mv, designer)$
  $dep_r(PR1, get\_vhf, designer)$
  $dep_r(PR2, get\_mv, designer)$
  $aut_r(PCD, update, designer)$
  $dep_r(PD, update\_disk, designer)$
  $aut_r(PRDL, report, designer)$
  $dep_r(PRPF, make\_report, designer)$

- en ce qui concerne les relations de a-dépendance entre deux agents :
  $dep\_on_a(PV, PVR1, scan, designer)$
  $dep\_on_a(PV, PVR2, scan, designer)$
  $dep\_on_a(PV, PCD, scan, designer)$
  $dep\_on_a(PV, PRDL, scan, designer)$
  $dep\_on_a(PVR1, PR1, scan\_vhf, designer)$
  $dep\_on_a(PVR2, PR2, scan\_mv, designer)$
  $dep\_on_a(PCD, PD, update\_disk, designer)$
  $dep\_on_a(PRDL, PRPF, make\_report, designer)$

- en ce qui concerne les relations de r-dépendance entre deux agents :
  $dep\_on_r(PVR1, PCD, scan\_vhf, designer)$
  $dep\_on_r(PVR2, PCD, scan\_mv, designer)$
  $dep\_on_r(PR1, PCD, get\_vhf, designer)$
  $dep\_on_r(PR2, PCD, get\_mv, designer)$
  $dep\_on_r(PD, PCD, update\_disk, designer)$
  $dep\_on_r(PRPF, PCD, make\_report, designer)$

Par rapport à ces formules, nous voulons détailler les points suivants :

- en utilisant les définitions d'agent sujet, objet, tiers et source introduites dans la section 5.2.1, nous notons que dans ce système c'est *le concepteur* qui tient lieu d'agent sujet et d'agent source, car la description externe n'existe pas en tant que telle au sein des agents du système. C'est pour cette raison que toutes les formules présentées ci-dessus ont comme dernier terme *designer*, que nous associons au concepteur, et qui correspond dans notre modèle à l'agent dont les plans sont utilisés pour inférer les diverses formules ;

- ce système étant conçu selon l'approche RDP, l'organisation de la société est conçue selon l'approche statique. Etant donné que le système n'est pas ouvert, les agents n'ont pas le choix en ce qui concerne leurs interactions, car ces dernières sont codées en dur. Autrement dit, dans ce système l'hypothèse de bienveillance est valable, et les relations de dépendance sont en quelque sorte synonymes du comportement du système : si un agent dépend de l'autre, cet agent va interagir avec l'agent de qui il dépend, en demandant l'exécution d'une certaine tâche, et le second va l'accomplir sans se demander s'il doit coopérer avec le premier ou non ;

- en ce qui concerne les situations de dépendance, elles n'ont pas beaucoup de sens dans ce contexte, car les agents n'ont pas de représentation des autres. Si nous les calculons, en considérant la description externe implicite dans l'esprit du concepteur, nous n'aurons que des *UD*s.

B.3 Résultats de simulation

En considérant la description externe du système EB décrite dans l'exemple B.1, les divers réseaux de a-dépendances et r-dépendances de tous les agents du système calculés par le simulateur DEPNET sont présentés respectivement dans les exemples B.2 et B.3 suivants.

Exemple B.2 : Réseaux de a-dépendance du système EB.

```
  PV
----------  scan
         |----------  scan:=scan_vhf(), scan_mv(),
                    |        update(), report().
                    |----------  scan_vhf
                               |**********  PVR1
                               |          |----------
                               |  scan_mv
                               |**********  PVR2
                               |          |----------
                               |  update
                               |**********  PCD
                               |          |----------
                               |  report
                               |**********  PRDL
                                          |----------

  PVR1
----------  scan_vhf
         |----------  scan_vhf:=get_vhf(addr_vhf).
                    |----------  get_vhf
                               |----------  PR1
                                          |----------

  PVR2
----------  scan_mv
         |----------  scan_mv:=get_mv(addr_mv).
                    |----------  get_mv
                               |----------  PR2
                                          |----------

  PR1
----------  get_vhf(addr_vhf)
         |----------  get_vhf(addr_vhf):=read_vhf(addr_vhf).
                    |----------  A-AUTONOMOUS
                               |----------

  PR2
----------  get_mv(addr_mv)
         |----------  get_mv(addr_mv):=read_mv(addr_mv).
                    |----------  A-AUTONOMOUS
                               |----------

  PCD
----------  update
         |----------  update:=write_memory(curr_ind), update_disk().
                    |----------  update_disk
                               |----------  PD
                                          |----------
```

```
  PD
----------  update_disk
         |----------  update_disk:=write_disk(curr_ind).
                    |----------  A-AUTONOMOUS
                               |----------

  PRDL
----------  report
         |----------  report:=make_report(), print_report().
                    |----------  make_report
                               |----------  PRPF
                                          |----------

  PRPF
----------  make_report
         |----------  make_report:=calculate_report(curr_ind).
                    |----------  A-AUTONOMOUS
                               |----------
```

Exemple B.3 : Réseaux de r-dépendance du système EB.

```
  PV
----------  scan
         |----------  scan:=scan_vhf(), scan_mv(),
                    |        update(), report().
                    |----------  R-AUTONOMOUS
                               |----------

  PVR1
----------  scan_vhf
         |----------  scan_vhf:=get_vhf(addr_vhf).
                    |----------  addr_vhf
                               |----------  PCD
                                          |----------

  PVR2
----------  scan_mv
         |----------  scan_mv:=get_mv(addr_mv).
                    |----------  addr_mv
                               |----------  PCD
                                          |----------

  PR1
----------  get_vhf(addr_vhf)
         |----------  get_vhf(addr_vhf):=read_vhf(addr_vhf).
                    |----------  addr_vhf
                               |----------  PCD
                                          |----------
```

```
  PR2
----------  get_mv(addr_mv)
         |----------  get_mv(addr_mv):=read_mv(addr_mv).
                   |----------  addr_mv
                             |----------  PCD
                                       |----------

  PCD
----------  update
         |----------  update:=write_memory(curr_ind), update_disk().
                   |----------  R-AUTONOMOUS
                             |----------

  PD
----------  update_disk
         |----------  update_disk:=write_disk(curr_ind).
                   |----------  curr_ind
                             |----------  PCD
                                       |----------

  PRDL
----------  report
         |----------  report:=make_report(), print_report().
                   |----------  R-AUTONOMOUS
                             |----------

  PRPF
----------  make_report
         |----------  make_report:=calculate_report(curr_ind).
                   |----------  curr_ind
                             |----------  PCD
```

Nous pouvons remarquer que les réseaux de dépendances présentées ci-dessus correspondent bien aux relations de dépendance que nous avons attribuées à ce système dans la section B.2. Si l'utilisateur du système voulait savoir, par exemple, quelles sont les situations de dépendance du type *MBMD* de l'agent *PV* pour son but *scan*, le système donnerait le résultat présenté dans l'exemple B.4.

Exemple B.4 : Situations de dépendance du système EB.

```
-----------------------------------------------
 **** mutually believed mutual dependences ****
-----------------------------------------------

Type the name of the agent to calculate the dependence: PV

Type the goal of the network: scan
```

```
Regarding agent:(PVR1)
---------------

There is a unilateral dependence between these agents regarding this goal
Agent (PVR1) has not goal scan in his current goal list

Regarding agent:(PVR2)
---------------

There is a unilateral dependence between these agents regarding this goal
Agent (PVR2) has not goal scan in his current goal list

Regarding agent:(PR1)
---------------

Agents (PV) and (PR1) are independent regarding goal scan
Agent (PR1) has not goal scan in his current goal list

Regarding agent:(PR2)
---------------

Agents (PV) and (PR2) are independent regarding goal scan
Agent (PR2) has not goal scan in his current goal list

Regarding agent:(PCD)
---------------

There is a unilateral dependence between these agents regarding this goal
Agent (PCD) has not goal scan in his current goal list

Regarding agent:(PD)
---------------

Agents (PV) and (PD) are independent regarding goal scan
Agent (PD) has not goal scan in his current goal list

Regarding agent:(PRDL)
---------------

There is a unilateral dependence between these agents regarding this goal
Agent (PRDL) has not goal scan in his current goal list

Regarding agent:(PRPF)
---------------

Agents (PV) and (PRPF) are independent regarding goal scan
Agent (PRPF) has not goal scan in his current goal list
```

Par concision, nous ne présentons pas ici les situations de dépendance qui seraient calculées par les autres agents de ce système, car comme nous en avons discuté dans la section B.2, la description externe n'existe pas en tant que telle au sein des agents, et par conséquent il n'y a que des *UD*s.

Annexe C

Description de l'organisation du système MAVI

Dans cette annexe, nous utilisons notre modèle introduit dans le chapitre 5 pour décrire l'organisation du système MAVI [BD94c]. Dans la section C.1, nous présentons les caractéristiques principales de ce système. L'application de notre modèle pour le décrire est présentée dans la section C.2. Enfin, dans la section C.3 nous montrons les résultats de simulation de ce modèle en utilisant le simulateur DEPNET.

C.1 Description du système

L'objectif du système intégré de vision MAVI [BD94c] est de construire la description symbolique d'une scène, à partir des séquences d'images obtenues par une caméra. Ses composantes correspondent bien à la notion d'agent telle que nous l'avons introduite dans la section 2.3.1. En effet, les agents sont conçus selon le modèle ASIC [Boi93, BD94b] présenté dans la section 3.4.

Ce système correspond bien à un exemple de conception selon l'approche SMA décrit dans la section 2.2.1 :

- la notion de protocoles d'interaction y est prise en compte et les agents l'exploitent explicitement ;
- de même pour la notion d'organisation, les agents adoptent les buts les uns des autres selon les contraintes imposées pour cette organisation.

Par contre, quelques points que nous jugeons comme importants ne sont pas traités dans ce système :

- le système n'est pas ouvert[1], il n'est pas prévu l'introduction des nouveaux agents pendant la phase d'exécution du système ;

[1] Au moins dans le sens précis que nous donnons a ce terme, c'est-à-dire, que n'importe quel agent puisse entrer dans le système, sans que le concepteur doive changer son organisation.

- l'organisation est conçue selon l'approche statique, elle n'est pas formée dynamiquement par des agents eux-mêmes comme nous le proposons dans le cadre de nos travaux ;
- comme nous l'avons discuté dans la section 4.4, les agents de ce système appliquent une procédure d'adoption de buts terminale, non-personnelle, et fonctionnelle.

Néanmoins, nous croyons que ce système, parmi ceux dont nous avons connaissance, est celui qui s'approche le plus du modèle de SMA ouvert que nous traitons dans ce travail. En effet, nous allons montrer à la fin de cette section qu'avec quelques petites modifications, il peut correspondre totalement à nos idées présentées jusqu'ici.

Le système MAVI est composé de 4 agents qui sont exécutés de façon concurrente, représentés dans la figure C.1, dont les fonctions principales sont les suivantes :

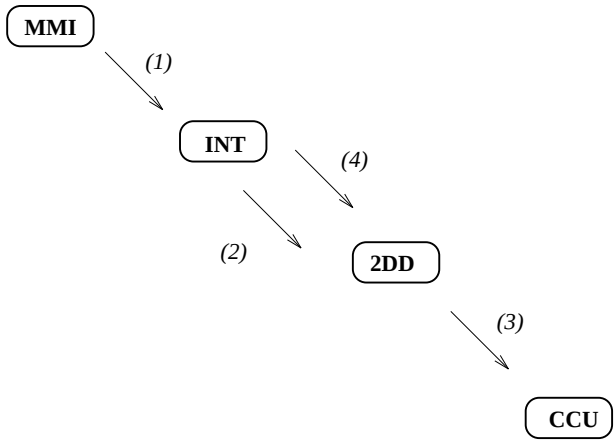


Figure C.1 : Système MAVI

1. *MMI* : interface homme-machine du système, qui permet à l'utilisateur de spécifier plusieurs actions, telles que la détermination de la région de l'image devant être exploitée et l'objet recherché dans l'image ;
2. *INT* : agent qui construit une description symbolique de l'image à partir de quelques segments et groupements perceptuels ;
3. *2DD* : agent qui construit la description de la scène en deux dimensions (2D) en termes de segments et de groupements perceptuels ;
4. *CCU* : agent qui récupère des images et contrôle les paramètres de la caméra, tels que son focus et sa vergence.

Les arcs représentés dans la figure C.1 montrent une partie d'un protocole d'interaction entre les agents du système. Une fois de plus, comme dans la section précédente, nous ne représentons que l'envoi d'une requête. Supposons que l'utilisateur demande au système, en interagissant avec l'agent MMI, de trouver un cube dans une scène particulière. Cet agent envoie une requête à l'agent INT pour accomplir cette tâche (1). Comme il faut à ce dernier les segments 2D pour construire une description symbolique, il demande à l'agent $2DD$ de construire ces segments (2). Pour sa part, l'agent $2DD$ a besoin d'une image pour accomplir cette tâche, et il demande à l'agent CCU de lui en envoyer une (3). Dans quelques situations, la description symbolique construite par l'agent INT peut présenter quelques inconsistances par rapport aux segments qui lui ont été envoyés par l'agent $2DD$. Dans ces cas, l'agent INT demande à l'agent $2DD$ de vérifier les segments que ce dernier lui avait envoyé (4).

C.2 Application du modèle

En utilisant notre définition de description externe, nous pouvons représenter les divers buts, actions, et plans de chaque agent de ce système, comme le montre l'exemple C.1.

Exemple C.1 : Description externe du système MAVI.

Agent	Buts	Actions	Ressources
MMI	get_cube	—	—
INT	find_cube	find_cube	—
		propose_cube	—
$2DD$	find_segments	find_segments	—
	verify_segments	verify_segments	—
		propose_segments	—
CCU	find_image	find_image	—

Agents	Plans		
MMI	get_cube()	:=	find_cube().
INT	find_cube()	:=	find_segments(), propose_cube(), verify_segments().
$2DD$	find_segments()	:=	find_image(), propose_segments().
	verify_segments()	:=	verify_segments().
CCU	find_image()	:=	find_image().

Nous n'avons pas besoin d'utiliser des ressources pour le décrire. Les diverses tâches sont une fois de plus représentées comme des actions, et les buts correspondent aux actions qui doivent être activées à un certain moment. Finalement,

il faut remarquer que cette description externe existe maintenant de fait au sein des agents du système[2].

Une fois de plus, en utilisant les diverses définitions de dépendance et autonomie introduites dans le chapitre 5, nous pouvons observer que les formules suivantes sont valables :

- en ce qui concerne l'autonomie et la dépendance vis-à-vis des actions :
  $dep_a(MMI, get\_cube, MMI)$
  $dep_a(INT, find\_cube, INT)$
  $dep_a(2DD, find\_segments, 2DD)$
  $aut_a(2DD, verify\_segments, 2DD)$
  $aut_a(CCU, find\_image, CCU)$

- en ce qui concerne les relations de a-dépendances entre deux agents :
  $dep\_on_a(MMI, INT, get\_cube, MMI)$
  $dep\_on_a(INT, 2DD, find\_cube, INT)$
  $dep\_on_a(2DD, CCU, find\_segments, 2DD)$

Par rapport à ces formules, nous voulons détailler les points suivants :

- en rappelant une fois de plus les définitions d'agent sujet, objet, tiers et source, chaque agent dans le système tient lieu d'agent sujet et d'agent source en ce qui concerne le calcul de ses propres dépendances, car la description externe existe de fait au sein des agents du système ;

- comme discuté dans le début de cette section, étant donné le fait que ce système n'est pas ouvert et que la procédure d'adoption de buts n'est pas instrumentale, la discussion faite dans la section précédente à propos de la liaison entre les relations de dépendance et le comportement du système reste elle aussi valable ;

- de même, comme les agents ne représentent pas les buts des autres, la discussion faite dans la section précédente à propos des situations de dépendance reste elle aussi valable, c'est-à-dire, on n'aura que des UDs.

A notre avis, pour que ce système puisse correspondre totalement au modèle de SMA traité dans ce travail, quelques modifications mineures sont nécessaires :

- en ce qui concerne l'ouverture du système, il faut implémenter un protocole d'introduction d'agents tel qu'il est décrit dans [BDB92], de façon à permettre aux agents eux-mêmes de mettre à jour leur description externe, ce qui n'est pas possible dans le modèle actuel ;

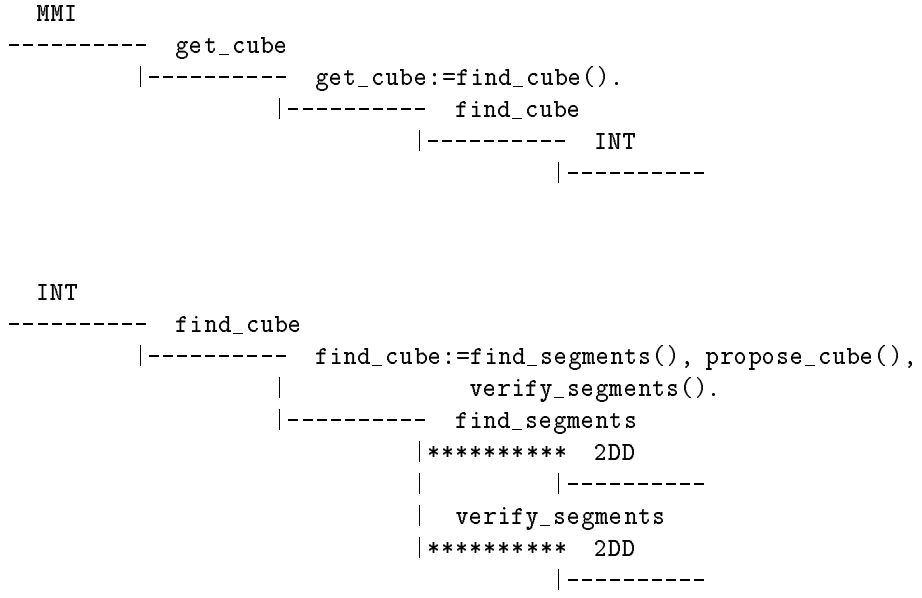
[2]Pour être précis, ce qui existe au sein des agents c'est une version simplifiée de notre modèle de description externe, correspondant seulement à la description des actions de chacun des agents.

- il faut représenter de façon plus complète au sein des agents notre notion de description externe. En particulier, si les agents représentent les buts des autres, la notion de situations de dépendance peut être mieux exploitée, en particulier pour détecter des dépendances mutuelles et réciproques et pour la formation dynamique de coalitions. Par conséquent, ce nouveau comportement revient à implémenter une procédure d'adoption de buts instrumentale chez les agents de ce système.

C.3 Résultats de simulation

En considérant la description externe du système MAVI décrite dans l'exemple C.1, les divers réseaux de a-dépendances de tous les agents du système calculés par le simulateur DEPNET sont présentés dans l'exemple C.2 suivant. Nous voulons rappeler que nous n'avons pas eu besoin d'utiliser des ressources pour décrire ce système, raison pour laquelle les réseaux de r-dépendances ne sont pas construits dans son simulation.

Exemple C.2 : Réseaux de a-dépendance du système MAVI.



```
  2DD
----------  find_segments
         |----------  find_segments:=find_image(),
         |                          propose_segments().
         |          |----------  find_image
         |                     |----------  CCU
         |                                |----------
         |  verify_segments
         |----------  verify_segments:=verify_segments().
                    |----------  A-AUTONOMOUS
                               |----------

  CCU
----------  find_image
         |----------  find_image:=find_image().
                    |----------  A-AUTONOMOUS
                               |----------
```

Une fois de plus, nous pouvons remarquer que les réseaux de dépendances présentées ci-dessus correspondent bien aux relations de dépendance que nous avons attribuées à ce système dans la section C.2.

Exemple C.3 : Situations de dépendance du système MAVI.

D-SIT		Agents			
moi	but	MMI	INT	$2DD$	CCU
MMI	*get_cube*	—	UD	—	—
INT	*find_cube*	—	—	UD	—
$2DD$	*find_segments*	—	—	—	UD
	verify_segments	—	—	—	—
CCU	*find_image*	—	—	—	—

Pour résumer, nous représentons les diverses situations de dépendance de ce système dans le exemple C.3 sous la forme d'un tableau, ne utilisant pas la sortie du système lui-même. Nous dénotons l'agent qui calcule ses situations de dépendance par le terme *moi* dans la première colonne[3]. Comme nous l'avons prévu dans la section C.2, nous n'avons que les UDs, car les agents ne représentent pas explicitement les buts les uns des autres.

[3]Dans le contexte de ce système, comme déjà expliqué dans la section C.2, ce calcul n'est fait par aucun agent.

Annexe D

Abréviations

AUT: autonome

DEP: dépendant

IA: intelligence artificielle

IAD: intelligence artificielle distribuée

IND: indépendance

LBMD: dépendance mutuelle localement reconnue

LBRD: dépendance réciproque localement reconnue

MD: dépendance mutuelle

MBMD: dépendance mutuelle mutuellement reconnue

MBRD: dépendance réciproque mutuellement reconnue

NG: but inexistant

NP: plan inexistant

RD: dépendance réciproque

RDP: résolution distribuée de problèmes

SMA: systèmes multi-agents

UD: dépendance unilatérale

Annexe E

Bibliographical remarks

We present below a list of significant bibliographical references that were either produced within the scope of this work or used as a theoretical or an implementation basis for it. The objectives of this list are twofold: (i) to enable an interested reader to obtain a detailed description of some particular issues addressed by our work and (ii) to enable non-French readers to obtain an overview of what is presented in this document.

For each reference in this list, we describe its main contents and we indicate to which chapter(s) it is particularly relevant.

[BD94b] Olivier Boissier and Yves Demazeau. ASIC: An architecture for social and individual control and its application to computer vision. In Yves Demazeau, Jean-Pierre Müller, and John Perram, editors, *Preproceedings of the 6th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, pages 107–118, Odense, Denmark, August 1994.

Contents: In this paper, the author presents an agent model, whose behaviour is defined by two control policies: an individual one and a social one. The individual control model is based on a multi-layer architecture, composed by a decision layer (responsible for defining which goal is to be satisfied), an adaptation layer (responsible for defining a plan to achieve the selected goal) and a command layer (responsible for the activation of the actions needed to accomplish the selected plan). The social control model defines the organisation of the agents and their interaction protocols, this organisation being conceived by a static approach. We have adopted a simplified version of the individual model as our agent model, described in chapter 3.

[Cas90] Cristiano Castelfranchi. Social power: A point missed in multi-agent, DAI and HCI. In Yves Demazeau and Jean-Pierre Müller, editors, *Decentralized A. I.*, pages 49–62. Elsevier Science Publishers B. V., Amsterdam, NL, 1990.

Contents: This paper presents an informal description of dependence and social power theory. Particularly, it stresses that influencing and goal adoption are the main basis upon which we can construct a theory to explain social interaction. We present these ideas in a different form in chapter 4.

[CC92] Rosaria Conte and Cristiano Castelfranchi. Mind is not enough: Precognitive bases of social interaction. In Nigel Gilbert, editor, *Proceedings of 1992 Symposium on Simulating Societies*, pages 93–110, Guildford, UK, April 1992.

Contents: This paper introduces the idea that both cognitive and precognitive notions must be taken into account in order to propose a bottom-up approach to social interactions. It presents the basic elements upon which such an approach can be constructed: social interference, social action and social interaction. We characterise these different elements in chapter 4.

[CMC92] Cristiano Castelfranchi, Maria Micelli, and Amedeo Cesta. Dependence relations among autonomous agents. In Eric Werner and Yves Demazeau, editors, *Decentralized A. I. 3*, pages 215–227. Elsevier Science Publishers B. V., Amsterdam, NL, 1992.

Contents: This paper presents a formal description of dependence and social power theory, based on the formal apparatus initially proposed by Cohen and Levesque in their work on intentions [CL87]. Particularly, the notions of resource and action dependence are formulated. We have not presented this formal description in this document, but we have borrowed some of its elements to propose our own formalisation of this model which is presented in chapter 5.

[CS92] Eleri Cardozo and Jaime Simão Sichman. DPSK+P user's manual - C++ interface. Technical report, DCA/FEE/UNICAMP, October 1992.

Contents: This report describes the main features of the DPSK+P system, an object-oriented distributed problem solving kernel which was developed to be used as a basis for integrating huge distributed intelligent applications. It enables a set of intelligent distributed agents to issue both communication and control actions over a network of workstations. The communication model is based on a metaphor of a single system-wide memory, which is structured using the object model, i.e., agents can dynamically create and instantiate shared classes and shared instances. This system was used to develop the communication layer of the MASENV platform, upon which we have implemented the DEPINT system described in chapter 11.

[CS95] Rosaria Conte and Jaime Simão Sichman. DEPNET: How to benefit from social dependence. *Journal of Mathematical Sociology (Special Issue on Sociological Algorithms)*, 1995. Forthcoming.

Contents: In this paper, we describe the use of our model in a social simulation context. We compare our modelling approach to social interaction based on complementarity with a pure utility based one, for instance that proposed by game theory. This comparison is presented in chapter 4. We also present some examples of simulations carried out using the DEPNET system, described in chapter 10.

[CSD93] Eleri Cardozo, Jaime Simão Sichman, and Yves Demazeau. Using the active object model to implement multi-agent systems. In *Proceedings of the 5th IEEE International Conference on Tools with Artificial Intelligence*, pages 70–77, Boston, USA, November 1993.

Contents: In this paper, we show how an active object model can be used as a basis for both integrating a single agent's internal mechanisms and inter-agent communication. We describe in this paper the implementation of the communication layer of the MASENV platform, upon which we have implemented the DEPINT system described in chapter 11.

[Dem95] Yves Demazeau. From interactions to collective behaviour in agent-based systems. In *Pre-proceedings of the invited lectures of the 1st European Conference on Cognitive Science*, St. Malo, France, March 1995.

Contents: In this paper, our approach to conceive multi-agent systems is described. This approach, based upon the decomposition between agents, environments, organisations and interaction, aims to facilitate software reuse, which is an aspect that may be used to differentiate multi-agent systems from distributed problem solving systems, as we discuss in chapter 2. The MASENV platform is presented, upon which we have built the DEPINT system, described in chapter 11.

[Dra93] Aldo Franco Dragoni. Distributed belief revision versus distributed truth maintenance: preliminary report. In Daniela D'Aloisi and Maria Miceli, editors, *Atti del 3zo Incontro del Gruppo AI*IA di Interesse Speciale su Inteligenza Artificiale Distribuita*, pages 64–73, Roma, Italia, December 1993. Roma:IP/CNR & ENEA.

Contents: In this paper, the author presents a model for belief revision in multi-agent systems. A criterion for choosing a context to be maintained, based on the reliability of information sources, is introduced. We have used this notion of credibility of information sources as one element to conceive our own criterion for choosing a context, described in chapter 8.

[Gas94] Graça Gaspar. *Modelação de Agentes Autónomos Inteligentes Integrados em Sociedades de Agentes.* Phd thesis, Departamento de Informática, Faculdade de Ciências da Universidade de Lisboa, Lisboa, Portugal, July 1994.

Contents: This thesis presents a complete formal model of autonomous intelligent agents. Concerning belief revision, a criterion for choosing a context to be maintained, based on the notion of authority of agents with respect to information topics, is introduced. We have used this notion of of information topics as another element to conceive our own criterion for choosing a context, described in chapter 8.

[Kon86] Kurt Konolige. *A Deduction Model of Belief.* Pitman Publishing, London, UK, 1986.

Contents: In this paper, a formal logic-based model characterising the notion of belief is presented. The most attractive aspect of this model is its sentential based semantics: the belief operator means that a certain formula belongs to the agent's belief set. This kind of semantics is suitable to the implementation of real agents, who are not omniscient and do not necessarily infer all of the conclusions from a set of premises. In chapter 8, we present the main features of this model and we use this formalism to prove that our social reasoning mechanism can detect agency level inconsistency, i.e., that agents may have either incomplete or incorrect information regarding one another.

[SCDC94] Jaime Simão Sichman, Rosaria Conte, Yves Demazeau, and Cristiano Castelfranchi. A social reasoning mechanism based on dependence networks. In Tony Cohn, editor, *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 188–192, Amsterdam, The Netherlands, August 1994. John Wiley & Sons Ltd.

Contents: This paper presents an earlier version of our formal model of dependence theory presented in chapter 5. We introduce for the first time the notion of dependence situations, which is a crucial aspect of our research. We also show some simulation results using the DEPNET system, described in chapter 10, which illustrate the possible dependence situations relating two agents.

[SD94a] Jaime Simão Sichman and Yves Demazeau. A first attempt to use dependence situations as a decision criterion for choosing partners in multi-agent systems. In *Proceedings of ECAI'94 Workshop on Decision Theory for DAI Applications*, Amsterdam, The Netherlands, August 1994.

Contents: In this paper, we present a decision criterion based on dependence situations which enables an agent to choose which possible partners he should address when he can not achieve a certain goal alone.

This criterion is based on two properties, the nature and the locality of the dependence. We propose a partial order to characterize the more advantageous dependence situations to be exploited by an agent. This criterion is detailed in chapter 7.

[SD94b] Jaime Simão Sichman and Yves Demazeau. Using class hierarchies to implement social reasoning in multi-agent systems. In *Anais do 11º Simpósio Brasileiro em Inteligência Artificial*, pages 27–41, Fortaleza, Brasil, October 1994. Sociedade Brasileira de Computação.

Contents: In this paper, we describe our implementation of the social reasoning mechanism, which was carried out using an object-oriented approach. Particularly, we propose two class hierarchies which implement respectively the external description and the dependence network of an agent. This description is presented in chapter 9.

[SD95a] Jaime Simão Sichman and Yves Demazeau. Exploiting social reasoning to deal with agency level inconsistency. In Victor Lesser, editor, *Proceedings of the 1st International Conference on Multi-Agent Systems*, pages 352–359, San Francisco, USA, June 1995. IEEE Computer Society Press.

Contents: In this paper, we show that in certain conditions, the fact that two agents infer different dependence situations relating one another for a same goal means that there is an agency level inconsistency, i.e., the information that they have about one another is either incomplete or incorrect. We present a detailed analysis of all possible coupled results of the social reasoning mechanisms of two agents (assuming that their planning mechanism is the same), where we make a relation between each coupled outcome and possible incomplete/incorrect information they may have about their actions and goals. This aspect of our research is presented in chapter 8. We have also presented in this paper the notion of goal situation, which has been incorporated in our formal model presented in chapter 5.

[SD95b] Jaime Simão Sichman and Yves Demazeau. Exploiting social reasoning to enhance adaptation in open multi-agent systems. In *Anais do 12º Simpósio Brasileiro em Inteligência Artificial*, Campinas, Brasil, October 1995. Sociedade Brasileira de Computação. Forthcoming.

Contents: In this paper, we show that our social reasoning mechanism can help an agent to adapt himself in open multi-agent systems. By open multi-agent systems, we mean that agents may either enter and leave the agency dynamically or change their current knowledge and beliefs during execution time. By taking into account these dynamic changes, an agent may be led to make different choices at different time instants, regarding the more advantageous partners to whom it

should address a proposal of cooperation. These results are presented in chapter 7.

[SDB92] Jaime Simão Sichman, Yves Demazeau, and Olivier Boissier. When can knowledge-based systems be called agents? In *Anais do 9º Simpósio Brasileiro em Inteligência Artificial*, pages 172–185, Rio de Janeiro, Brasil, October 1992. Sociedade Brasileira de Computação.

Contents: This paper presents a survey of the multi-agent systems domain, characterising its history, main research axes, etc. It proposes an earlier version of our agent model, whose updated version is presented in chapter 3. An updated version of this survey is presented in chapter 2.

[WJ94] Michael Wooldridge and Nicholas R. Jennings. Towards a theory of cooperative problem solving. In Yves Demazeau, Jean-Pierre Müller, and John Perram, editors, *Pre-proceedings of the 6th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, pages 15–26, Odense, Denmark, August 1994.

Contents: In this paper, a formal model of a cooperative problem solving process is presented. This model is based on the idea of coalition formation. We have adopted this model as a basis for the dynamic conception of agents' organisations, described in chapter 3.

Bibliographie

[ACG86] Ahuja (Sudhir), Carriero (Nicholas) et Gelernter (David). – LINDA and friends. *Computer*, vol. 19, n° 8, August 1986, pp. 26–34.

[AH86] Agha (Gul) et Hewitt (Carl E.). – Concurrent programming using actors: Exploiting large-scale parallelism. *In: Readings in Distributed Artificial Intelligence*, éd. par Bond (Alan H.) et Gasser (Les), pp. 398–407. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1986.

[Arm93] Armitage (James). – *Process Guide for the Domain-Specific Software Architectures (DSSA) Process Life Cycle.* – Technical report n° CMU/SEI-93-SR-21, Software Engineering Institute, Carnegie Mellon University, 1993.

[Axe84] Axelrod (R.). – *The Evolution of Cooperation.* – New York, Basic Books, 1984.

[BAF+87] Bisiani (Roberto), Alleva (F.), Forin (A.), Lerner (R.) et Bauer (M.). – The architecture of the AGORA environment. *In: Distributed Artificial Intelligence*, éd. par Huhns (Michael N.), pp. 99–107. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1987.

[Bat94] Bates (Joseph). – The role of emotions in believable agents. *Communications of the ACM*, vol. 37, n° 7, July 1994, pp. 122–125.

[BD94a] Boddy (Mark) et Dean (Thomas L.). – Deliberation scheduling for problem solving in time-constrained environments. *Artificial Intelligence*, vol. 67, 1994, pp. 245–285.

[BD94b] Boissier (Olivier) et Demazeau (Yves). – ASIC: An architecture for social and individual control and its application to computer vision. *In: Pre-proceedings of the 6th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, éd. par Demazeau (Yves), Müller (Jean-Pierre) et Perram (John), pp. 107–118. – Odense, Denmark, August 1994.

[BD94c] Boissier (Olivier) et Demazeau (Yves). – MAVI: A multi-agent system for visual integration. *In: Proceedings of the IEEE Interna-*

tional Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 731–738. – Las Vegas, USA, October 1994.

[BD94d] Bussmann (Stefan) et Demazeau (Yves). – An agent model combining reactive and cognitive capabilities. *In : Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS'94)*. – Munich, Germany, September 1994.

[BDA95] Baeijs (Christof), Demazeau (Yves) et Alvares (Luis O.). – Application des systèmes multi-agents à la généralisation cartographique. *In : Actes des 3èmes Journées Francophones Intelligence Artificielle Distribuée & Systèmes Multi-Agents*, éd. par LIA (Université de Savoie). – Chambéry, France, March 1995.

[BDB92] Berthet (Sabine), Demazeau (Yves) et Boissier (Olivier). – Knowing each other better. *In : Proceedings of the 11th International Workshop on Distributed Artificial Intelligence*, pp. 23–42. – Glen Arbor, MI, February 1992.

[Ber93] Bernstein (Philip A.). – *Middleware: An Architecture for Distributed System Services.* – Technical report n° CRL 93/6, Digital Equipment Corporation, March 1993.

[BG88] Bond (Alan H.) et Gasser (Les) (édité par). – *Readings in Distributed Artificial Intelligence.* – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1988.

[BJV94] Bijnens (Stijn), Joosen (Wouter) et Verbaeten (Pierre). – Language constructs for coordination in an agent space. *In : Pre-proceedings of the 6th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, éd. par Demazeau (Yves), Müller (Jean-Pierre) et Perram (John), pp. 65–77. – Odense, Denmark, August 1994.

[BLR93] Bates (Joseph), Loyall (A. Brian) et Reilly (W. Scott). – An architecture for action, emotion and social behavior. *In : Artificial Social Systems*, éd. par Castelfranchi (Cristiano) et Werner (Eric), pp. 55–68. – Berlin, DE, Springer-Verlag, 1993.

[Boi93] Boissier (Olivier). – *Problème du Contrôle dans un Système Integré de Vision. Utilisation d'un Système Multi-Agents.* – Grenoble, France, Thèse de Doctorat, Institut National Polytechnique de Grenoble, January 1993.

[Bou92] Bouron (Thierry). – *Structures de Communication et d'Organisation pour la Cooperation dans un Univers Multi-Agents.* – Paris, France, Thèse de Doctorat, Université Paris VI, November 1992.

[Bou95] Bourgine (Paul). – Models of self-teaching agents and the emergence of the symbolic level. *In: Pre-proceedings of the invited lectures of the 1st European Conference on Cognitive Science.* – St. Malo, France, March 1995.

[BRCHV95] Bordini (Rafael H.), Rocha Costa (Antônio C.), Hübner (Jomi) et Viccari (Rosa M.). – Linguistic support for agent migration. *In: Proceedings of the 1st International Conference on Multi-Agent Systems*, éd. par Lesser (Victor). p. 441. – San Francisco, USA, June 1995.

[Bre90] Brewka (Gerhard). – Belief revision in a framework for default reasoning. *In: The Logic of Theory Change*, éd. par Fuhrmann (André) et Morreau (Michael), pp. 206–222. – Berlin, DE, Springer-Verlag, 1990.

[Bro86] Brooks (Rodney A.). – A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, vol. 2, n° 1, March 1986, pp. 435–453.

[BS92] Burmeister (Birgit) et Sundermeyer (Kurt). – Cooperative problem-solving guided by intentions and perception. *In: Decentralized A. I. 3*, éd. par Werner (Eric) et Demazeau (Yves), pp. 77–92. – Amsterdam, NL, Elsevier Science Publishers B. V., 1992.

[CAM94] Coelho (Helder), Antunes (Luis) et Moniz (Luis). – On agent design rationale. *In: Anais do 11º Simpósio Brasileiro em Inteligência Artificial.* Sociedade Brasileira de Computação, pp. 43–58. – Fortaleza, Brasil, October 1994.

[Car87] Cardozo (Eleri). – *DPSK: A Kernel for Distributed Problem Solving.* – Pittsburgh, PE, Phd Thesis, CAED, Carnegie Mellon University, January 1987.

[Car92] Carley (Kathleen). – Organizational learning and personnel turnover. *Organization Science*, vol. 3, n° 1, 1992, pp. 20–46.

[Cas90] Castelfranchi (Cristiano). – Social power: A point missed in multi-agent, DAI and HCI. *In: Decentralized A. I.*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 49–62. – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[Cas92] Castelfranchi (Cristiano). – No more cooperation, please! in search of the social structure of verbal interaction. *In: Communication from an Artificial Intelligence Perspective: Theoretical and Applied Issues*, éd. par Ortony (Andrew), Slack (Jon) et Stock (Oliviero), pp. 205–227. – Berlin, DE, Springer-Verlag, 1992.

[Cas95] Castelfranchi (Cristiano). – Guarantees for autonomy in cognitive agent architecture. *In: Intelligent Agents*, éd. par Wooldridge (Michael J.) et Jennings (Nicholas R.), pp. 56–70. – Berlin, DE, Springer-Verlag, 1995.

[CC92] Conte (Rosaria) et Castelfranchi (Cristiano). – Mind is not enough: Precognitive bases of social interaction. *In: Proceedings of 1992 Symposium on Simulating Societies*, éd. par Gilbert (Nigel), pp. 93–110. – Guildford, UK, April 1992.

[CC93a] Conte (Rosaria) et Castelfranchi (Cristiano). – Norms as mental objects: From normative beliefs to normative goals. *In: Pre-proceedings of the 5th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, éd. par Ghedira (Khaled) et Sprumont (François). – Neuchâtel, Switzerland, August 1993.

[CC93b] Correa (Milton) et Coelho (Helder). – Around the architectural agent approach to model conversations. *In: Pre-proceedings of the 5th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, éd. par Ghedira (Khaled) et Sprumont (François). – Neuchâtel, Switzerland, August 1993.

[CC94] Conte (Rosaria) et Castelfranchi (Cristiano). – Minds and norms: Types of normative reasoning. *In: Proceedings of the 2nd Italia-USA Workshop on Knowledge, Belief and Strategic Interaction*, éd. par Bicchieri (C.). – Cambridge University Press.

[CC95] Castelfranchi (Cristiano) et Conte (Rosaria). – Agent exchange value and emerging organizational structure. – 1995. Submitted to Computational and Mathematical Organization Theory Journal.

[CCC93] Costa (Antônio Carlos da Rocha), Castilho (Josée Mauro Volkmer) et Cláudio (Dalcídio Moraes). – Functional processes and functional roles in societies of computing agents. *In: Anais do 10º Simpósio Brasileiro em Inteligência Artificial.* Sociedade Brasileira de Computação. – Porto Alegre, Brasil, October 1993.

[CCZ94] Carle (Patrice), Collinot (Anne) et Zeghal (Karim). – Concevoir des organisations: La méthode Cassiopée. *In: Actes de la 3ème Journée Systèmes Multi-Agents du PRC-GDR Intelligence Artificielle*, éd. par Demazeau (Yves) et Collinot (Anne). – Paris, France, December 1994.

[CD90] Campbell (John A.) et D'Inverno (Mark P.). – Knowledge interchange protocols. *In: Decentralized A. I.*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 63–80. – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[CD93] Crowley (James L.) et Demazeau (Yves). – Principles and techniques for sensor data fusion. *Signal Processing*, vol. 32, n° 1-2, May 1993, pp. 5–27.

[CFO+93] Cardozo (Eleri), Fernandes (João F. M.), Oliveira (Gina M. B.), Sichman (Jaime Simão) et Demazeau (Yves). – Decentralized organizations for artificial intelligence: Some implementation issues. *In: Proceedings of COMDEX/SUCESU-SP South America'93.* – São Paulo, Brasil, August 1993.

[CGHH89] Cohen (Paul R.), Greenberg (Michael L.), Hart (David M.) et Howe (Adele E.). – Trial by fire: Understanding the design requirements for agents in complex environments. *AI Magazine*, 1989, pp. 32–48. – Fall.

[CGR92] Coelho (Helder), Gaspar (Graça) et Ramos (Isabel). – Experiments on achieving communication in communities of autonomous agents. *In: Proceedings of the IFIP WG 8.3 Working Conference on Decision Support Systems: Experiments and Expectations.* – Fontainbleau, France, July 1992.

[Cha81] Chandrasekan (B.). – Natural and social system methaphors for distributed problem solving: Introduction to the issue. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 11, n° 1, January 1981, pp. 1–5.

[CHB94] Costa (Antônio Carlos da Rocha), Hübner (Jomi Fred) et Bordini (Rafael Heitor). – On entering an open society. *In: Anais do 11° Simpósio Brasileiro em Inteligência Artificial.* Sociedade Brasileira de Computação, pp. 535–546. – Fortaleza, Brasil, October 1994.

[Cho93] Cholvy (Laurence). – Proving theorems in a multi-source environment. *In: Proceedings of the 13th International Joint Conferennce on Artificial Intelligence*, éd. par Bajcsy (Ruzena). pp. 66–71. – Chambery, France, August 1993.

[Cho94] Cholvy (Laurence). – Fusion de sources d'informations contradictoires ordonnées en fonction des thèmes. *Revue IA*, 1994. – À paraître.

[CL83] Corkill (Daniel D.) et Lesser (Victor R.). – The distributed vehicle monitoring testbed: A tool for investigating distributed problem solving networks. *AI Magazine*, vol. 4, n° 3, Fall 1983, pp. 15–33.

[CL87] Cohen (Philip R.) et Levesque (Hector J.). – Intention = choice + commitment. *In: Proceedings of the 6th National Conference on Artificial Intelligence.* pp. 410–415. – Seattle, WA, July 1987.

[CL90] Cohen (Philip R.) et Levesque (Hector J.). – Persistence, intention and commitment. *In: Intentions in Communication*, éd. par Cohen (Philip R.), Morgan (Jerry) et Pollack (Martha E.), chap. 3, pp. 33–69. – Cambridge, MA, MIT Press, 1990.

[CM92] Cesta (Amedeo) et Miceli (Maria). – How to benefit from others: Dependence theory and agent architecture. *In: Atti del 2do Incontro del Gruppo AI*IA di Interesse Speciale su Inteligenza Artificiale Distribuita*, éd. par Conte (Rosaria) et Fiorentino (Sila), pp. 95–104. – Roma, Italia, November 1992.

[CM93] Cesta (Amedeo) et Miceli (Maria). – In search of help: Strategic social knowledge and plans. *In: Proceedings of the 12th International Workshop on Distributed Artificial Intelligence.* – Hidden Valley, PA, May 1993.

[CMC91] Conte (Rosaria), Miceli (Maria) et Castelfranchi (Cristiano). – Limits and levels of cooperation: Disentangling various types of prosocial interaction. *In: Decentralized A. I. 2*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 147–157. – Amsterdam, NL, Elsevier Science Publishers B. V., 1991.

[CMC92] Castelfranchi (Cristiano), Micelli (Maria) et Cesta (Amedeo). – Dependence relations among autonomous agents. *In: Decentralized A. I. 3*, éd. par Werner (Eric) et Demazeau (Yves), pp. 215–227. – Amsterdam, NL, Elsevier Science Publishers B. V., 1992.

[Coh91] Cohen (Paul R.). – A survey of the eighth national conference on artificial intelligence: Pulling together or pulling apart? *AI Magazine*, 1991, pp. 17–41. – Spring.

[Con92] Conte (Rosaria). – Three-party dependence and rational communication. *In: Atti del 2do Incontro del Gruppo AI*IA di Interesse Speciale su Inteligenza Artificiale Distribuita*, éd. par Conte (Rosaria) et Fiorentino (Sila), pp. 105–114. – Roma, Italia, November 1992.

[CS92] Cardozo (Eleri) et Sichman (Jaime Simão). – *DPSK+P User's Manual - C++ Interface.* – Technical report, DCA/FEE/UNICAMP, October 1992.

[CS95] Conte (Rosaria) et Sichman (Jaime Simão). – DEPNET: How to benefit from social dependence. *Journal of Mathematical Sociology*, vol. 19, n° 2, 1995. – Special Issue on Sociological Algorithms.

[CSD93] Cardozo (Eleri), Sichman (Jaime Simão) et Demazeau (Yves). – Using the active object model to implement multi-agent systems. *In: Proceedings of the 5th IEEE International Conference on Tools with Artificial Intelligence*, pp. 70–77. – Boston, USA, November 1993.

[CW92] Chang (Man Kit) et Woo (Carson C.). – SANP: A communication level protocol for negotiations. *In: Decentralized A. I. 3*, éd. par Werner (Eric) et Demazeau (Yves), pp. 31–54. – Amsterdam, NL, Elsevier Science Publishers B. V., 1992.

[DBK94a] Demazeau (Yves), Boissier (Olivier) et Koning (Jean-Luc). – Interaction protocols in distributed artificial intelligence and their use to control robot vision systems. *In: Proceedings of the 6th. International Conference on Artificial Intelligence and Information-Control Systems.* – Smolenice, Slovaquia, September 1994.

[DBK94b] Demazeau (Yves), Boissier (Olivier) et Koning (Jean-Luc). – Using interaction protocols to control vision systems. *In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics.* – San Antonio, USA, October 1994.

[DD92] Drogoul (Alexis) et Dubreil (Christophe). – Eco-Problem Solving model: Results of the N-Puzzle. *In: Decentralized A. I. 3*, éd. par Werner (Eric) et Demazeau (Yves), pp. 283–295. – Amsterdam, NL, Elsevier Science Publishers B. V., 1992.

[Dem91] Demazeau (Yves). – Coordination patterns in multi-agent worlds: Applications to computer vision and robotics. *In: Proceedings of the IEE Coloquium on Intelligent Agents.* – Savoy Place, February 1991.

[Dem93] Demazeau (Yves). – La plate-forme PACO et ses applications. *In: Actes de la 2ème Journée Systèmes Multi-Agents du PRC-GDR Intelligence Artificielle*, éd. par Demazeau (Yves) et Collinot (Anne). – Montpellier, France, December 1993.

[Dem94] Demazeau (Yves). – Distributed AI and multi-agent systems. – Lindø Center for Applied Mathematics, University of Odense, DK, August 1994. Lecture Notes.

[Dem95] Demazeau (Yves). – From interactions to collective behaviour in agent-based systems. *In: Pre-proceedings of the invited lectures of the 1st European Conference on Cognitive Science.* – St. Malo, France, March 1995.

[dK86] de Kleer (Johan). – An assumption-based TMS. *Artificial Intelligence*, vol. 28, n° 2, 1986.

[DLC87] Durfee (Edmund H.), Lesser (Victor R.) et Corkill (Daniel D.). – Coherent cooperation among communicating problem solvers. *IEEE Transactions on Computers*, vol. 36, n° 11, November 1987, pp. 1275–1291.

[DM90a] Demazeau (Yves) et Müller (Jean-Pierre) (édité par). – *Decentralized A. I.* – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[DM90b] Demazeau (Yves) et Müller (Jean-Pierre). – Decentralized artificial intelligence. *In: Decentralized A. I.*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 3–13. – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[DM91] Demazeau (Yves) et Müller (Jean-Pierre). – From reactive to intentional agents. *In: Decentralized A. I. 2*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 3–10. – Amsterdam, NL, Elsevier Science Publishers B. V., 1991.

[Doy79] Doyle (Jon). – A truth maintenance system. *Artificial Intelligence*, vol. 12, n° 2, 1979.

[Doy92] Doyle (Jon). – Rationality and its role in reasoning. *Computational Intelligence*, vol. 8, n° 376-409, 1992.

[DR94] Durfee (Edmund H.) et Rosenschein (Jeffrey S.). – Distributed problem solving and multi-agent systems: Comparisons and examples. *In: Proceedings of the 13th International Workshop on Distributed Artificial Intelligence.* – Seattle, WA, 1994.

[Dra93] Dragoni (Aldo Franco). – Distributed belief revision versus distributed truth maintenance: preliminary report. *In: Atti del 3zo Incontro del Gruppo AI*IA di Interesse Speciale su Inteligenza Artificiale Distribuita*, éd. par D'Aloisi (Daniela) et Miceli (Maria). Roma:IP/CNR & ENEA, pp. 64–73. – Roma, Italia, December 1993.

[Dro93] Drogoul (Alexis). – *De la Simulation Multi-Agents à la Résolution Collective de Problèmes: Une Ètude de l'Èmergence de Structures d'Organisation dans les Systèmes Multi-Agents.* – Paris, France, Thèse de Doctorat, Université Paris VI, November 1993.

[EHRLR80] Erman (Lee D.), Hayes-Roth (Frederick), Lesser (Victor R.) et Reddy (D. Raj). – The hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty. *Computing Surveys*, vol. 12, n° 2, June 1980, pp. 213–253.

[ELS92] Etzioni (Oren), Lesh (Neal) et Segal (Richard B.). – *Building Softbots for UNIX (preliminary report).* – Technical report n° 93-9-01, Seattle, WA, Department of Computer Science and Engineering, University of Washington, November 1992.

[EM88] Engelmore (Robert S.) et Morgan (Anthony) (édité par). – *Blackboard Systems.* – Reading, Addison-Wesley, 1988.

[ER87] Euzenat (Jerôme) et Rechenmann (François). – Maintenance de la verité dans les systèmes à base de connaissances centrée-objet. *In: Actes du 6ème Congrès Reconnaissance de Formes et Intelligence Artificielle*. pp. 1096–1109. – Antibes, France, November 1987.

[ER93] Ephrati (Eitan) et Rosenschein (Jeffrey S.). – Multi-agent planning as a dynamic search for social consensus. *In: Proceedings of the 13th International Joint Conferennce on Artificial Intelligence*, éd. par Bajcsy (Ruzena). pp. 423–429. – Chambery, France, August 1993.

[EW94] Etzioni (Oren) et Weld (Daniel). – A softbot-based interface to the internet. *Communications of the ACM*, vol. 37, n° 7, July 1994, pp. 72–76.

[FD90] Fraichard (Thierry) et Demazeau (Yves). – Motion planning in a multi-agent world. *In: Decentralized A. I.*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 137–153. – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[Fer92] Ferguson (Innes A.). – *Touringmachines: An Architecture for Dynamic, Rational, Mobile Agents.* – Cambridge, UK, Phd Thesis, Computer Laboratory, University of Cambridge, October 1992.

[Fer93] Fernandez (Jose L.). – *A Taxonomy of Coordination Mechanisms Used in Real-Time Software Based on Domain Analysis.* – Technical report n° CMU/SEI-93-TR-34, Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, December 1993.

[FG87] Farreny (Henri) et Ghallab (Malik). – *Eléments d'Intelligence Artificielle.* – Paris, FR, Hermès, 1987.

[FG91] Ferber (Jacques) et Gasser (Les). – Intelligence artificielle distribuée. – EC2, Avignon, FR, 1991. Tutorial Notes of the 11th Conference on Expert Systems and their Applications (Avignon'91).

[FJ91] Ferber (Jacques) et Jacopin (Eric). – The framework of eco-problem solving. *In: Decentralized A. I. 2*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 181–193. – Amsterdam, NL, Elsevier Science Publishers B. V., 1991.

[Fox81] Fox (Michael S.). – An organizational view of distributed systems. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 11, n° 1, January 1981, pp. 70–80.

[Gal73] Galbraith (J.). – *Designing Complex Organizations.* – Reading, Addison-Wesley, 1973.

[Gal88] Galliers (Julia Rose). – *A Theoretical Framework for Computer Models of Cooperative Dialogue, Acknowledging Multi-Agent Conflict.* – London, UK, Phd Thesis, Department of Psychology, Open University, September 1988.

[Gal91] Galliers (Julia Rose). – Modelling autonomous belief revision in dialogue. *In: Decentralized A. I. 2*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 231–243. – Amsterdam, NL, Elsevier Science Publishers B. V., 1991.

[Gas91] Gaspar (Graça). – Communication and belief changes in a society of agents: Towards a formal model of an autonomous agent. *In: Decentralized A. I. 2*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 245–255. – Amsterdam, NL, Elsevier Science Publishers B. V., 1991.

[Gas92a] Gasser (Les). – Boundaries, identity and aggregation: Plurality issues in multiagent systems. *In: Decentralized A. I. 3*, éd. par Werner (Eric) et Demazeau (Yves), pp. 199–212. – Amsterdam, NL, Elsevier Science Publishers B. V., 1992.

[Gas92b] Gasser (Les). – An overview of DAI. *In: DAI: Theory and Praxis*, éd. par Avouris (N. M.) et Gasser (Les), pp. 179–195. – Amsterdam, NL, Kluwer Academic Publishers, 1992.

[Gas94] Gaspar (Graça). – *Modelação de Agentes Autónomos Inteligentes Integrados em Sociedades de Agentes.* – Lisboa, Portugal, Tese de Doutoramento, Departamento de Informática, Faculdade de Ciências da Universidade de Lisboa, July 1994.

[GBH87] Gasser (Les), Braganza (Carl) et Herman (Nava). – MACE: A flexible testbed for distributed AI research. *In: Distributed Artificial Intelligence*, éd. par Huhns (Michael N.), pp. 119–152. – London, UK, Pitman Publishing, 1987.

[GD93] Gmytrasiewicz (Piotr J.) et Durfee (Edmund H.). – Reasoning about other agents: Philosophy, theory and implementation. *In: Preproceedings of the 5th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, éd. par Ghedira (Khaled) et Sprumont (François). – Neuchâtel, Switzerland, August 1993.

[GH94] Glance (Natalie) et Huberman (Bernardo). – La dynamique des dilemmes sociaux. *Pour la Science*, vol. 199, May 1994, pp. 26–31.

[GS90] Grosz (Barbara) et Sidner (Candace L.). – Plans for discourse. *In: Intentions in Communication*, éd. par Cohen (Philip R.), Morgan (Jerry) et Pollack (Martha E.), chap. 20, pp. 417–444. – Cambridge, MA, MIT Press, 1990.

[HB91] Huhns (Michael N.) et Bridgeland (David M.). – Multiagent truth maintenance. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, n° 6, 1991, pp. 1437–1445.

[HDL92] Hassoun (Mouna), Demazeau (Yves) et Laugier (Christian). – Motion control for a car-like robot: Potential field and multi-agent approaches. *In: Proceedings of the 1992 International Conference on Intelligent Robots and Systems (IROS'92).* – Raleigh, USA, August 1992.

[Hew86] Hewitt (Carl E.). – Offices are open systems. *In: Readings in Distributed Artificial Intelligence*, éd. par Bond (Alan H.) et Gasser (Les), pp. 312–329. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1986.

[Hew93] Hewitt (Carl E.). – Some requirements for mobile distributed telecomputing architecture. *In: Artificial Social Systems*, éd. par Castelfranchi (Cristiano) et Werner (Eric), pp. 259–270. – Berlin, DE, Springer-Verlag, 1993.

[HH94] Huberman (Bernardo A.) et Hogg (T.). – *Communities of Practice: Performance and Evolution.* – Technical report, Xerox Palo Alto Research Center, 1994.

[HI91] Hewitt (Carl E.) et Inman (Jeff). – DAI betwixt and between: From "intelligent agents" to open systems science. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, n° 6, 1991, pp. 1409–1419.

[Hin62] Hintikka (J.). – *Knowledge and Belief.* – Ithaca, NY, Cornell University Press, 1962.

[IGY92] Ishida (Toru), Gasser (Les) et Yokoo (M.). – Organization self-design of distributed production systems. *IEEE Transactions on Knowledge and Data Engineering*, vol. 4, n° 2, April 1992, pp. 123–184.

[Jen92] Jennings (Nicholas R.). – Towards a cooperation knowledge level for collaborative problem solving. *In: Proceedings of the 10th European Conference on Artificial Intelligence*, éd. par Neumann (Bernd). pp. 224–228. – Vienna, Austria, August 1992.

[Jen93] Jennings (Nicholas R.). – Commitments and conventions: The foundation of coordination in multi-agent systems. *The Knowledge Engineering Review*, vol. 8, n° 3, 1993, pp. 223–250.

[JP93] Jennings (Nicholas R.) et Pople (Jeff A.). – Design and implementation of ARCHON's coordination module. *In: Proceedings of the Workshop on Cooperating Knowledge-Based Systems.* – Keele, UK, 1993.

[JVA+93] Jennings (Nicholas R.), Varga (L. Z.), Aarnts (R. P.), Fuchs (J.) et Sharek (P.). – Transforming standalone expert systems into a community of cooperating agents. *International Journal of Engineering Applications of Artificial Intelligence*, vol. 6, n° 4, 1993, pp. 317–331.

[JW92] Jennings (Nicholas R.) et Wittig (Thys). – ARCHON: Theory and praxis. *In: DAI: Theory and Praxis*, éd. par Avouris (N. M.) et Gasser (Les), pp. 179–195. – Amsterdam, NL, Kluwer Academic Publishers, 1992.

[Kar95] Karam (Myriam). – *Description d'Agents vers une Programmation Orientée Agent.* – Grenoble, France, Rapport de DEA Informatique, ENSIMAG, INPG, 1995.

[KB95] Kreiling (Amy K.) et Baker (Frank). – *NCSA Mosaic Handbook.* – Englewood Cliffs, NJ, Prentice-Hall International, January 1995.

[KDEQ95] Koning (Jean-Luc), Demazeau (Yves), Esfandiari (Babak) et Quinqueton (Jöel). – Quelques perspectives d' utilisation des langages et protocoles d'intéraction dans le contexte des telecommunications. *In: Actes des 3èmes Journées Francophones Intelligence Artificielle Distribuée & Systèmes Multi-Agents*, éd. par LIA (Université de Savoie). – Chambéry, France, March 1995.

[KN80] Konolige (Kurt) et Nilsson (Nils J.). – Multiple-agent planning systems. *In: Proceedings of the 1st National Conference on Artificial Intelligence*, pp. 138–141. – Stanford, CA, August 1980.

[Kno90] Knoke (David). – *Political Networks: The Structural Perspective.* – Cambridge, UK, Cambridge University Press, 1990, *Structural Analysis in the Social Sciences Series.*

[Kon86] Konolige (Kurt). – *A Deduction Model of Belief.* – London, UK, Pitman Publishing, 1986.

[KP93] Konolige (Kurt) et Pollack (Martha E.). – A representationalist theory of intention. *In: Proceedings of the 13th International Joint Conferennce on Artificial Intelligence*, éd. par Bajcsy (Ruzena). pp. 390–395. – Chambery, France, August 1993.

[Kra93] Kraus (Sarit). – Agent contracting in non-collaborative environments. *In: Proceedings of the 11th National Conference on Artificial Intelligence*, pp. 243–248. – Washington, DC, August 1993.

[LCD94] Lizotte (Sylvain) et Chaib-Draa (Brahim). – Un modèle de raisonnement sur autrui basé sur les graphes causaux. *In: Actes des 2èmes Journées Francophones Intelligence Artificielle Distribuée &*

Systèmes Multi-Agents, éd. par PLEIAD, pp. 319–330. – Voiron, France, May 1994.

[LCD95] Lizotte (Sylvain) et Chaib-Draa (Brahim). – Coordination en situations non familières. *In: Actes des 3èmes Journées Francophones Intelligence Artificielle Distribuée & Systèmes Multi-Agents*, éd. par LIA (Université de Savoie), pp. 255–266. – Chambéry, France, March 1995.

[LCN90] Levesque (Hector J.), Cohen (Philip R.) et Nunes (José H. T.). – On acting together. *In: Proceedings of the 8th National Conference on Artificial Intelligence*. pp. 94–99. – Boston, August 1990.

[Len75] Lenat (Douglas B.). – BEINGs: Knowledge as interacting experts. *In: Readings in Distributed Artificial Intelligence*, éd. par Bond (Alan H.) et Gasser (Les), pp. 161–168. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1975.

[Lev90] Levi (Paul). – Architectures of individual and distributed autonomous agents. *In: Intelligent Autonomous Agents 2*, éd. par Kanade (T.), Groen (F. C. A.) et Hertzberger (L. O.), pp. 315–324. – Amsterdam, The Netherlands, IAS, 1990.

[Lom93] Lomborg (Bjorn). – Game theory vs multiple agents: The iterated prisoner's dilemma. *In: Artificial Social Systems*, éd. par Castelfranchi (Cristiano) et Werner (Eric), pp. 69–93. – Berlin, DE, Springer-Verlag, 1993.

[LR57] Luce (R. D.) et Raiffa (H.). – *Games and Decisions: Introduction and Critical Survey.* – John Wiley & Sons Ltd., 1957.

[LS95] Le Strugeon (Emmanuelle). – *Une Méthodologie d'Auto-Adaptation d'un Système Multi-Agents Cognitifs.* – Valenciennes, France, Thèse de Doctorat, Université de Valenciennes et du Haut-Cambrésis, January 1995.

[Mae94] Maes (Pattie). – Agents that reduce work and information overload. *Communications of the ACM*, vol. 37, n° 7, July 1994, pp. 30–40.

[Mal87] Malone (Thomas W.). – Modeling coordination in organizations and markets. *In: Readings in Distributed Artificial Intelligence*, éd. par Bond (Alan H.) et Gasser (Les), pp. 151–158. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1987.

[Mar93] Marsh (Stephen). – Trust and reliance in multi-agent systems: A preliminary report. *In: Artificial Social Systems*, éd. par Castelfranchi (Cristiano) et Werner (Eric), pp. 94–114. – Berlin, DE, Springer-Verlag, 1993.

[MCF+94] Mitchell (Tom), Caruana (Rich), Freitag (Dayne), McDermott (John) et Zabovski (David). – Experience with a learning personal assistant. *Communications of the ACM*, vol. 37, n° 7, July 1994, pp. 80–91.

[MCM83] Michalski (Ryszard S.), Carbonell (Jaime G.) et Mitchell (Tom M.) (édité par). – *Machine Learning: An Artificial Intelligence Approach.* – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1983.

[Min89] Minsky (Naftaly H.). – *The Imposition of Protocols over Open Distributed Systems.* – Technical report n° LCSR-TR-154, New Jersey, USA, Laboratory for Computer Science Research, Rutgers University, April 1989.

[MIT90] Maruichi (Takeo), Ichikawa (Masaki) et Tokoro (Mario). – Modeling autonomous agents and their groups. *In: Decentralized A. I.*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 215–234. – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[MJ89] Mason (Cyndy L.) et Johnson (Rowland R.). – DATMS: A framework for distributed assumption based reasoning. *In: Distributed Artificial Intelligence vol II*, éd. par Gasser (Les) et Huhns (Michael N.), pp. 293–317. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1989.

[MJO94] Malheiro (Benedita), Jennings (Nicholas R.) et Oliveira (Eugénio). – Belief revision in multi-agent systems. *In: Proceedings of the 11th European Conference on Artificial Intelligence*, éd. par Cohn (Tony). pp. 294–298. – Amsterdam, The Netherlands, August 1994.

[MR89] Minsky (Naftaly H.) et Rozenshtein (David). – Controllable delegation: An exercise in law-governed systems. *In: Proceedings of the ACM Conference on Object-Oriented Programming Systems, Languages and Applications*, éd. par Meyrowitz (Norman). pp. 371–380. – New Orleans, LA, October 1989.

[MS88] Martins (João P.) et Shapiro (Stuart C.). – A model for belief revision. *Artificial Intelligence*, vol. 35, n° 2, 1988.

[New82] Newell (Allen). – The knowledge level. *Artificial Intelligence*, vol. 18, 1982, pp. 87–127.

[Nil94] Nilsson (Nils J.). – Teleo-reactive programs for agent control. *Journal of Artificial Intelligence*, vol. 1, 1994, pp. 139–158.

[NWM93] Nicol (John R.), Wilkes (Thomas) et Manola (Frank A.). – Obejct orientation in heterogeneous distributed computing systems. *IEEE Computer*, June 1993, pp. 57–67.

[OAA+94] Oquendo (Flavio), Alloui (Ilham), Arbaoui (Selma), Debord (Bernard), Latrous (Safia) et Tassart (Guy). – SCALE/ALPS: Un générateur de systèmes multi-agents pour la construction d'environnements centrés processus assistées par ordinateur. *In: Actes des 2èmes Journées Francophones Intelligence Artificielle Distribuée & Systèmes Multi-Agents*, éd. par PLEIAD, pp. 249–261. – Voiron, France, May 1994.

[OC91] Oliveira (Eugénio) et Camacho (Rui). – A shell for cooperating expert systems. *Expert Systems*, vol. 8, n° 2, May 1991, pp. 75–85.

[OCR91] Oliveira (Eugénio), Camacho (Rui) et Ramos (C.). – A multi-agent environment in robotics. *Robotica*, vol. 9, 1991, pp. 431–440.

[OD94] Occello (Michel) et Demazeau (Yves). – Building real time agents using parallel blackboards and its use for mobile robotics. *In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics.* – San Antonio, TX, October 1994.

[OPP94] Overgaard (Lars), Petersen (Henrik G.) et Perram (John W.). – Motion planning for an articulated robot: A multi-agent approach. *In: Pre-proceedings of the 6th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, éd. par Demazeau (Yves), Müller (Jean-Pierre) et Perram (John), pp. 171–182. – Odense, Denmark, August 1994.

[Ost87] Ostrovsky (R.). – *Holmes-1, A Prolog-Based Reasoning Maintenance System for Collecting Information From Multiple Experts.* – Berlin, DE, Springer-Verlag, 1987, *Lecture Notes in Computer Science*, volume 286.

[OZT90] Oquendo (Flavio), Zucker (Jean-Daniel) et Tassart (Guy). – Support for software tool integration and process centered software engineering environments. *In: Proceedings of the 3th International Conference on Software Engineering and its Applications.* – Toulouse, France, December 1990.

[Par90] Parunak (H. van Dyke). – Toward a formal model of inter-agent control. *In: Proceedings of the 10th International Workshop on Distributed Artificial Intelligence.* – Bandera, TX, October 1990.

[PBS93] Populaire (Philippe), Boissier (Olivier) et Sichman (Jaime Simão). – Description et implementation de protocoles de communication en univers multi-agents. *In: Actes des 1ères Journées Francophones Intelligence Artificielle Distribuée & Systèmes Multi-Agents*, éd. par IRIT, pp. 241–252. – Toulouse, France, April 1993.

[Pit92] Pitrat (Jacques). – La symbiose entre l'intelligence artificielle et la science cognitive. *Technique et Science Informatiques*, vol. 11, n° 2, 1992, pp. 9–24.

[PLE92] PLEIAD. – Vers une taxinomie du vocabulaire pour les systèemes multi-agents. *In: Actes de la 1ère Journée Systèmes Multi-Agents du PRC-GDR Intelligence Artificielle*, éd. par Demazeau (Yves) et Ferber (Jacques). – Nancy, France, November 1992.

[Pol86] Pollack (Martha). – A model of plan inference that distinguishes between the beliefs of actors and observers. *In: Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics.*

[Pol90] Pollack (Martha). – Plans as complex mental attitudes. *In: Intentions in Communication*, éd. par Cohen (Philip R.), Morgan (Jerry) et Pollack (Martha E.), chap. 5, pp. 77–103. – Cambridge, MA, MIT Press, 1990.

[RB92] Reilly (W. Scott) et Bates (Joseph). – *Building Emotional Agents.* – Technical report n° CMU-CS-92-143, Pittsburgh, PA, School of Computer Science, Carnegie Mellon University, May 1992.

[RG85] Rosenschein (Jeffrey S.) et Genesereth (Michael R.). – Deals among rational agents. *In: Proceedings of the 9th International Joint Conferennce on Artificial Intelligence.* pp. 91–99. – Los Angeles, CA, 1985.

[RG95] Rao (Anand S.) et Georgeff (Michael P.). – BDI agents: From theory to practice. *In: Proceedings of the 1st International Conference on Multi-Agent Systems*, éd. par Lesser (Victor). pp. 312–319. – San Francisco, USA, June 1995.

[RGS92] Rao (Anand S.), Georgeff (Michael P.) et Sonenberg (Elizabeth A.). – Social plans: A preliminary report. *In: Decentralized A. I. 3*, éd. par Werner (Eric) et Demazeau (Yves), pp. 57–76. – Amsterdam, NL, Elsevier Science Publishers B. V., 1992.

[RK86] Rosenschein (Jeffrey S.) et Kaelbling (Leslie P.). – The synthesis of digital machines with provable epistemic properties. *In: Theoretical Aspects of Reasoning About Knowledge*, éd. par Halpern (Joseph Y.), pp. 83–98. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1986.

[Rod94] Rodriguez (Miguel). – *Modélisation d'un Agent Autonome: Approche Constructiviste de l'Architecture de Contrôle et de la Représentation de Connaissances.* – Neuchâtel, Suisse, Thèse de Doctorat, Université de Neuchâtel, September 1994.

[RW91] Russel (Stuart) et Wefald (Eric). – Principles of metareasoning. *Artificial Intelligence*, vol. 49, 1991, pp. 361–395.

[Rym91] Rymer (J. R.). – Common object request broker - OMG's new standard for distributed object management. *Network Monitor*, vol. 6, n° 9, September 1991, pp. 3–27.

[RZ94] Rosenschein (Jeffrey S.) et Zlotkin (Gilad). – *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers.* – Cambridge, MA, MIT Press, 1994.

[SB83] Stefik (Mark) et Bobrow (Daniel G.). – Object-oriented programming: Themes and variations. *AI Magazine*, vol. 6, n° 4, Winter 1983, pp. 40–64.

[SC81] Steeb (Randall) et Cammarata (Stephanie). – Architectures for distributed air-traffic control. *In: Readings in Distributed Artificial Intelligence*, éd. par Bond (Alan H.) et Gasser (Les), pp. 90–101. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1981.

[SCDC94] Sichman (Jaime Simão), Conte (Rosaria), Demazeau (Yves) et Castelfranchi (Cristiano). – A social reasoning mechanism based on dependence networks. *In: Proceedings of the 11th European Conference on Artificial Intelligence*, éd. par Cohn (Tony). pp. 188–192. – Amsterdam, The Netherlands, August 1994.

[SD94a] Sichman (Jaime Simão) et Demazeau (Yves). – A first attempt to use dependence situations as a decision criterion for choosing partners in multi-agent systems. *In: Proceedings of ECAI'94 Workshop on Decision Theory for DAI Applications.* – Amsterdam, The Netherlands, August 1994.

[SD94b] Sichman (Jaime Simão) et Demazeau (Yves). – Using class hierarchies to implement social reasoning in multi-agent systems. *In: Anais do 11º Simpósio Brasileiro em Inteligência Artificial.* Sociedade Brasileira de Computação, pp. 27–41. – Fortaleza, Brasil, October 1994.

[SD95a] Sichman (Jaime Simão) et Demazeau (Yves). – Exploiting social reasoning to deal with agency level inconsistency. *In: Proceedings of the 1st International Conference on Multi-Agent Systems*, éd. par Lesser (Victor). pp. 352–359. – San Francisco, USA, June 1995.

[SD95b] Sichman (Jaime Simão) et Demazeau (Yves). – Exploiting social reasoning to enhance adaptation in open multi-agent systems. *In: Anais do 12º Simpósio Brasileiro em Inteligência Artificial.* Sociedade Brasileira de Computação. – Campinas, Brasil, October 1995. Forthcoming.

[SDB92] Sichman (Jaime Simão), Demazeau (Yves) et Boissier (Olivier). – When can knowledge-based systems be called agents? *In: Anais do 9º Simpósio Brasileiro em Inteligência Artificial.* Sociedade Brasileira de Computação, pp. 172–185. – Rio de Janeiro, Brasil, October 1992.

[Sea69] Searle (John). – *Speach Acts.* – Cambridge University Press, 1969.

[Sho91] Shoham (Yoav). – Agent oriented programming. *Artificial Intelligence*, vol. 60, n° 1, 1991, pp. 51–92.

[Sin91] Singh (Munindar P.). – Towards a formal theory of communication for multiagent systems. *In: Proceedings of the 12th International Joint Conferennce on Artificial Intelligence.* – Sydney, Australia, 1991.

[SJ92] Sichman (Jaime Simão) et Jeszenszky (Paul Jean Ettienne). – EB: A telemetry data acquisition system. *In: Proceedings of the 7th. Symposium on Microcomputer and Microprocessor Applications*, pp. 307–315. – Budapest, Hungary., April 1992.

[Smi80] Smith (Reid G.). – The contract net protocol: High-level communication and control is a distributed problem solver. *IEEE Transactions on Computers*, vol. 29, n° 12, December 1980, pp. 1104–1113.

[ST92] Shoham (Yoav) et Tennenholtz (Moshe). – On the systhesis of useful social laws for artificial agent societies (preliminary report). *In: Proceedings of the 10th National Conference on Artificial Intelligence*, pp. 276–281. – San Jose, CA, July 1992.

[Ste90] Steels (Luc). – Cooperation between distributed agents through self-organization. *In: Decentralized A. I.*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 175–196. – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[Ste93] Stefanini (Marie-Hélène). – Vers une architecture multi-agents pour le traitement automatique des langues naturelles. *In: Actes des 1ères Journées Francophones Intelligence Artificielle Distribuée & Systèmes Multi-Agents*, éd. par IRIT, pp. 153–165. – Toulouse, France, April 1993.

[Tay92] Taylor (C. J.). – A status report on open distributed processing. *First-Class (Object Management Group Newsletter)*, vol. 2, n° 2, 1992, pp. 11–13.

[Tho93] Thomas (Sarah R.). – *PLACA: An Agent Oriented Programming Language.* – Stanford, CA, Phd thesis, department of Computer Science, Stanford University, September 1993, 294–298p.

[Tic92] Tichy (Walter F.). – Programming-in-the-large: Past, present and future. *In: Proceedings of the 14th International Conference on Software Engineering.* – Melbourne, Australia, 1992.

[Tok93] Tokoro (Mario). – *The society of objects.* – Technical report n° SCSL-TR-93-018, Tokyo, Japan, Sony Computer Science Laboratory Inc., December 1993.

[vM90a] von Martial (Frank). – Coordination of plans in multiagent worlds by taking advantage of the favor relation. *In: Proceedings of the 10th International Workshop on Distributed Artificial Intelligence.* – Bandera, TX, 1990.

[vM90b] von Martial (Frank). – Interactions among autonomous planning agents. *In: Decentralized A. I.*, éd. par Demazeau (Yves) et Müller (Jean-Pierre), pp. 105–119. – Amsterdam, NL, Elsevier Science Publishers B. V., 1990.

[Wai94a] Wainer (Jacques). – Reasoning about another agent through empathy. *In: Anais do 11° Simpósio Brasileiro em Inteligência Artificial.* Sociedade Brasileira de Computação, pp. 17–25. – Fortaleza, Brasil, October 1994.

[Wai94b] Wainer (Jacques). – Yet another semantics of goal and goal priorities. *In: Proceedings of the 11th European Conference on Artificial Intelligence*, éd. par Cohn (Tony). pp. 267–273. – Amsterdam, The Netherlands, August 1994.

[Wer89] Werner (Eric). – Cooperating agents: A unified theory of communication and social structure. *In: Distributed Artificial Intelligence vol II*, éd. par Gasser (Les) et Huhns (Michael N.), pp. 3–36. – San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1989.

[WG95] Wavish (Peter) et Graham (Michael). – Roles, skills and behaviour: A situated action approach to organising systems of interacting agents. *In: Intelligent Agents*, éd. par Wooldridge (Michael J.) et Jennings (Nicholas R.), pp. 371–385. – Berlin, DE, Springer-Verlag, 1995.

[WJ94] Wooldridge (Michael) et Jennings (Nicholas R.). – Towards a theory of cooperative problem solving. *In: Pre-proceedings of the 6th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, éd. par Demazeau (Yves), Müller (Jean-Pierre) et Perram (John), pp. 15–26. – Odense, Denmark, August 1994.

[Woo92] Wooldridge (Michael). – *The Logical Modelling of Computational Multi-Agent Systems.* – Manchester, UK, Phd Thesis, Department of Computation, Manchester Metropolitan University, September 1992.

[YM93] Yu (Eric S. K.) et Mylopoulos (John). – An actor dependency model of organizational work with application to business process reengineering. *In: Proceedings of the Conference on Organizational Computing Systems (COOCS'93)*, éd. par Kaplan (Simon). pp. 258–268. – Milpitas, CA, November 1993.

[ZR90] Zlotkin (Gilad) et Rosenschein (Jeffrey S.). – Negotiation and conflict resolution in non-cooperative domains. *In: Proceedings of the 8th National Conference on Artificial Intelligence*, pp. 100–105. – Boston, August 1990.

[ZR93] Zlotkin (Gilad) et Rosenschein (Jeffrey S.). – A domain theory for task oriented negotiation. *In: Proceedings of the 13th International Joint Conferennce on Artificial Intelligence*, éd. par Bajcsy (Ruzena). pp. 416–422. – Chambery, France, August 1993.

Résumé

Cette thèse présente le modèle d'un mécanisme de raisonnement social fondé sur la théorie de la dépendance. Ce modèle permet à un agent de raisonner sur autrui et plus particulièrement de calculer ses relations et situations de dépendance. Un agent est dépendant d'un autre si celui-ci peut l'aider/l'empêcher d'atteindre un de ses buts. Nous considérons notre mécanisme de raisonnement social comme un composant essentiel pour la conception d'agents artificiels réellement autonomes, évoluant dans un univers multi-agents ouvert. La notion d'ouverture désigne la capacité d'ajouter ou de retirer dynamiquement dans le système des agents. Comme dans ces systèmes l'organisation des agents ne peut pas être spécifiée pendant la phase de conception, la résolution coopérative de problèmes est fondée sur la formation dynamique de coalitions. Dans ce contexte, des agents doivent être capables de s'adapter aux changements dynamiques du système, en particulier en évaluant pendant la phase de résolution si leurs buts sont réalisables et si leurs plans sont exécutables. Comme nous ne supposons pas que les agents soient bienveillants, notre modèle fournit un critère pour évaluer les partenaires les plus susceptibles d'accepter une proposition de coalition. Enfin, comme dans ces systèmes des agents n'ont pas généralement une représentation complète et correcte les uns des autres, notre modèle leur permet de détecter une inconsistance au niveau de la société et de choisir un contexte à être maintenue. Nous avons implémenté ce mécanisme de raisonnement social en utilisant une programmation orientée objet. Nous l'avons utilisé pour développer deux applications, le simulateur DEPNET et le système DEPINT, qui illustrent son utilisation selon deux perspectives scientifiques différentes. D'une part, selon une perspective de simulation sociale, notre modèle fournit un outil informatique permettant l'analyse et la prédiction des divers schémas intéressants d'interaction sociale, et l'évaluation du pouvoir social des agents. D'autre part, selon une perspective de résolution de problèmes, notre modèle peut être utilisé pour concevoir dynamiquement l'organisation des agents dans un contexte de systèmes multi-agents ouverts.

Mots clés : raisonnement social, raisonnement sur autrui, intelligence artificielle distribuée, systèmes multi-agents, intégration de systèmes, systèmes ouverts.

Abstract

This thesis presents the model of a social reasoning mechanism based on dependence theory. This model enables an agent to reason about the others, in particular to calculate his dependence relations and dependence situations. An agent is said to be dependent on another if the latter can help/prevent him to achieve one of his goals. We consider our social reasoning mechanism as an essential building block for the design of really autonomous artificial agents, which are immersed in an open multi-agent world. By open, we mean that agents may enter or leave the agency at any moment. In such systems, as the organisation of the agents can not be conceived at design time, the cooperative problem solving paradigm is based on dynamic coalition formation. In this context, agents must be able to adapt themselves to dynamically changing conditions, by evaluating at execution time if their goals are achievable and if their plans are feasible. As we do not suppose that agents are benevolent, our model proposes a criterion to evaluate which partners are more susceptible to accept a proposition of coalition. Finally, as in these kind of systems agents usually do not have a complete and correct representation of each other, our model helps them to detect an agency level inconsistency and to choose a context to be maintained. We have implemented our social reasoning mechanism using an object-oriented approach, and we have used it to develop two applications, the DEPNET simulator and the DEPINT system, which illustrate respectively its usage in two different scientific perspectives. On one hand, concerning social simulation, our model provides a computational tool for the analysis and prediction of the occurrence of several interesting patterns of social interactions, and for the evaluation of the agents' social power. On the other hand, with respect to problem solving, our model can be used to design dynamic agents' organizations in a context of open multi-agent systems.

Keywords : social reasoning, reasoning about the others, distributed artificial intelligence, multi-agent systems, system integration, open systems.

Resumo

Esta tese propõe um modelo de um mecanismo de raciocínio social baseado na teoria da dependência. Tal modelo permite a um agente raciocinar sobre os outros, em particular calcular suas relações e situações de dependência. Um agente depende de um outro se este último pode lhe facilitar/prejudicar a obtenção de um de seus objetivos. Nós consideramos nosso mecanismo de raciocínio social como um elemento fundamental para a concepção de agentes artificiais realmente autônomos, que estejam imersos num universo multi-agentes aberto. A noção de abertura denota o fato de que os agentes podem entrar ou sair do sistema em qualquer momento. Em tais sistemas, como a organização dos agentes não pode ser especificada durante a sua fase de concepção, o modelo cooperativo de resolução de problemas se baseia na formação dinâmica de coligações. Neste contexto, os agentes devem ser capazes de se adaptar a mudanças dinâmicas, avaliando durante a fase de execução se seus objetivos são realizáveis e se seus planos são executáveis. Como nós não supomos que os agentes sejam benevolentes, nosso modelo propõe um critério para avaliar os agentes mais susceptíveis a aceitar uma proposta de coligação. Finalmente, como em tais sistemas os agentes normalmente não têm uma representação completa e correta uns dos outros, nosso modelo lhes permite detectar uma inconsistência global e escolher un contexto a ser mantido. Nós implementamos este mecanismo de raciocínio social utilizando uma programação orientada a objetos, e o utilisamos para desenvolver duas aplicações, o simulador DEPNET e o sistema DEPINT, que ilustram o seu interesse sob duas perspectivas científicas diversas. Por um lado, considerando uma perspectiva de simulação social, tal modelo fornece uma ferramenta computacional para a análise e predição de diversos tipos de interações sociais interessantes e para a avaliação do poder social dos agentes. Por outro lado, no que diz respeito à resolução de problemas, nosso modelo pode ser utilizado para a concepção dinâmica de organizações de agentes num contexto de sistemas multi-agentes abertos.

Palavras chave : raciocínio social, raciocínio sobre os outros, inteligência artificial distribuída, sistemas multi-agentes, integração de sistemas, sistemas abertos.

Riassunto

Questa tesi presenta un modello di meccanismo di raggionamento sociale basato sulla teoria della dipendenza. Tale modello abilita un agente a raggionare su altri agenti, in particolare ad estimare le sue relazioni e situazioni di dipendenza. Un agente é detto dipendente da un altro agente se quest'último può aiutare/impedire il raggiungimento dei suoi scòpi. Riteniamo che il nostro meccanismo di raggionamento sociale sia un essenziale contributo alla teoria degli agenti artificiali autonomi in un mondo multi-agenti aperto. Il concetto de l'apertura significa che gli agenti possono entrare e/o uscire dal sistema in qualsiasi istante. In tali sistemi, come l'organizzazione degli agenti non può essere specificata durante la fasi di proggetazione, il modello cooperativo di risoluzione dei problemi si basa nella formazione dinamica di coalizione. In questo contexto, gli agenti debbono essere capabili di adattarsi a modifique dinamique, valutando durante la fasi di esecuzione si i suoi scòpi sono raggiungibili e si i suoi piani sono eseguibili. Come noi non supponiamo che gli agenti siano benevolenti, il nostro modello propone un criterio di valutazione degli agenti piu suscettibili ad accettare una proposta di coalizione. Finalmente, come in questi sistemi gli agenti non hanno una rappresentazione completa e corretta nei confronti degli altri agenti, questo modello permette un agente de scoprire una inconsistenza globale e di scegliere un contexto ad essere seguito. Noi abbiamo applicato il modello di programmazione orientata ad oggetti per implementare questo meccanismo de raggionamento sociale e l'abbiamo utilizzato per sviluppare le due applicazione, il simulatore DEPNET ed il sistema DEPINT, che illustrano sua importanza su due prospettive scientifique diverse. Da un canto, considerando una prospettiva di simulazione sociale, il modello descritto permette uno studio approfondito dei diversi tipi di interazioni sociali e la valutazione del potere sociale degli agenti. D'altro canto, il modello da noi proposto dá la possibilitá di organizzare dinamicamente gli agenti nel contexto dei sistemi multi-agenti aperti.

Parole chiavi : raggionamento sociale, raggionamento su gli altri, intelligenza artificiale distribuita, sistemi multi-agenti, integrazione dei sistemi, sistemi aperti.

Jaime Simão Sichman **Du Raisonnement Social Chez les Agents** *Une Approche Fondée sur la Théorie de la Dépendance*