Systèmes multi-agent pour le diagnostic pluri-disciplinaire

Julien Dumont

Ecole Nationale Supérieure des Mines
SAINT-ETIENNE

N° d'ordre : 2011EMSE0603

THÈSE

présentée par

Julien DUMONT

pour obtenir le grade de
Docteur de l'École Nationale Supérieure des Mines de Saint-Étienne

Spécialité : INFORMATIQUE

SYSTEME MULTI-AGENTS POUR LE DIAGNOSTIC PLURI-DISCIPLINAIRE

soutenue à Saint-Etienne, le 02 Décembre 2010

Membres du jury

Président :	Prénom NOM	Grade, Établissement, Ville
Rapporteurs :	Prénom NOM	Grade, Établissement, Ville
	Prénom NOM	Grade, Établissement, Ville
Examinateur(s) :	Prénom NOM	Grade, Établissement, Ville
Directeur de thèse :	Olivier BOISSIER	Professeur, ENS Mines Saint-Etienne
Co-encadrant	Guy VINZANT	ECAI Saint-Etienne
Invité(s) éventuel(s):	Prénom NOM	Grade, Établissement, Ville

Spécialités doctorales :	**Responsables :**
SCIENCES ET GENIE DES MATERIAUX	J. DRIVER Directeur de recherche – Centre SMS
MECANIQUE ET INGENIERIE	A. VAUTRIN Professeur – Centre SMS
GENIE DES PROCEDES	G. THOMAS Professeur – Centre SPIN
SCIENCES DE LA TERRE	B. GUY Maître de recherche – Centre SPIN
SCIENCES ET GENIE DE L'ENVIRONNEMENT	J. BOURGOIS Professeur – Centre SITE
MATHEMATIQUES APPLIQUEES	E. TOUBOUL Ingénieur – Centre G2I
INFORMATIQUE	O. BOISSIER Professeur – Centre G2I
IMAGE, VISION, SIGNAL	JC. PINOLI Professeur – Centre CIS
GENIE INDUSTRIEL	P. BURLAT Professeur – Centre G2I
MICROELECTRONIQUE	Ph. COLLOT Professeur – Centre CMP

Enseignants-chercheurs et chercheurs autorisés à diriger des thèses de doctorat (titulaires d'un doctorat d'État ou d'une HDR)

AVRIL	Stéphane	MA	Mécanique & Ingénierie	CIS
BATTON-HUBERT	Mireille	MA	Sciences & Génie de l'Environnement	SITE
BENABEN	Patrick	PR 2	Sciences & Génie des Matériaux	CMP
BERNACHE-ASSOLANT	Didier	PR 0	Génie des Procédés	CIS
BIGOT	Jean-Pierre	MR	Génie des Procédés	SPIN
BILAL	Essaïd	DR	Sciences de la Terre	SPIN
BOISSIER	Olivier	PR 2	Informatique	G2I
BOUCHER	Xavier	MA	Génie Industriel	G2I
BOUDAREL	Marie-Reine	MA	Génie Industriel	DF
BOURGOIS	Jacques	PR 0	Sciences & Génie de l'Environnement	SITE
BRODHAG	Christian	DR	Sciences & Génie de l'Environnement	SITE
BURLAT	Patrick	PR 2	Génie industriel	G2I
COLLOT	Philippe	PR 1	Microélectronique	CMP
COURNIL	Michel	PR 0	Génie des Procédés	DF
DAUZERE-PERES	Stéphane	PR 1	Génie industriel	CMP
DARRIEULAT	Michel	IGM	Sciences & Génie des Matériaux	SMS
DECHOMETS	Roland	PR 1	Sciences & Génie de l'Environnement	SITE
DESRAYAUD	Christophe	MA	Mécanique & Ingénierie	SMS
DELAFOSSE	David	PR 1	Sciences & Génie des Matériaux	SMS
DOLGUI	Alexandre	PR 1	Génie Industriel	G2I
DRAPIER	Sylvain	PR 2	Mécanique & Ingénierie	SMS
DRIVER	Julian	DR	Sciences & Génie des Matériaux	SMS
FEILLET	Dominique	PR 2	Génie Industriel	CMP
FOREST	Bernard	PR 1	Sciences & Génie des Matériaux	CIS
FORMISYN	Pascal	PR 1	Sciences & Génie de l'Environnement	SITE
FORTUNIER	Roland	PR 1	Sciences & Génie des Matériaux	SMS
FRACZKIEWICZ	Anna	DR	Sciences & Génie des Matériaux	SMS
GARCIA	Daniel	CR	Génie des Procédés	SPIN
GIRARDOT	Jean-Jacques	MR	Informatique	G2I
GOEURIOT	Dominique	MR	Sciences & Génie des Matériaux	SMS
GRAILLOT	Didier	DR	Sciences & Génie de l'Environnement	SITE
GROSSEAU	Philippe	MR	Génie des Procédés	SPIN
GRUY	Frédéric	MR	Génie des Procédés	SPIN
GUY	Bernard	MR	Sciences de la Terre	SPIN
GUYONNET	René	DR	Génie des Procédés	SPIN
HERRI	Jean-Michel	PR 2	Génie des Procédés	SPIN
INAL	Karim	MR	Microélectronique	CMP
KLÖCKER	Helmut	DR	Sciences & Génie des Matériaux	SMS
LAFOREST	Valérie	CR	Sciences & Génie de l'Environnement	SITE
LERICHE	Rodolphe	CR	Mécanique et Ingénierie	SMS
LI	Jean-Michel	EC (CCI MP)	Microélectronique	CMP
LONDICHE	Henry	MR	Sciences & Génie de l'Environnement	SITE
MOLIMARD	Jérôme	MA	Mécanique et Ingénierie	SMS
MONTHEILLET	Frank	DR 1 CNRS	Sciences & Génie des Matériaux	SMS
PERIER-CAMBY	Laurent	PR1	Génie des Procédés	SPIN
PIJOLAT	Christophe	PR 1	Génie des Procédés	SPIN
PIJOLAT	Michèle	PR 1	Génie des Procédés	SPIN
PINOLI	Jean-Charles	PR 0	Image, Vision, Signal	CIS
STOLARZ	Jacques	CR	Sciences & Génie des Matériaux	SMS
SZAFNICKI	Konrad	CR	Sciences & Génie de l'Environnement	DF
THOMAS	Gérard	PR 0	Génie des Procédés	SPIN
VALDIVIESO	François	MA	Sciences & Génie des Matériaux	SMS
VAUTRIN	Alain	PR 0	Mécanique & Ingénierie	SMS
VIRICELLE	Jean-Paul	MR	Génie des procédés	SPIN
WOLSKI	Krzysztof	DR	Sciences & Génie des Matériaux	SMS
XIE	Xiaolan	PR 1	Génie industriel	CIS

Glossaire :

PR 0	Professeur classe exceptionnelle
PR 1	Professeur 1ère catégorie
PR 2	Professeur 2ème catégorie
MA(MDC)	Maître assistant
DR	Directeur de recherche
Ing.	Ingénieur
MR(DR2)	Maître de recherche
CR	Chargé de recherche
EC	Enseignant-chercheur
IGM	Ingénieur général des mines

Centres :

SMS	Sciences des Matériaux et des Structures
SPIN	Sciences des Processus Industriels et Naturels
SITE	Sciences Information et Technologies pour l'Environnement
G2I	Génie Industriel et Informatique
CMP	Centre de Microélectronique de Provence
CIS	Centre Ingénierie et Santé

Dernière mise à jour le : 5 février 2010

À mon grand-père.

Remerciements

Mes premières pensées se tournent vers mon grand-père, un homme généreux et avide de connaissances, qui après quelques discussions avec mes parents a su les convaincre de l'intérêt d'un Doctorat. Je regrette juste qu'il ne soit plus là aujourd'hui pour voir le résultat de ces trois années de travail. En honneur à sa mémoire, je tiens à lui dédier ce présent manuscrit.

Les encouragements de mon grand-père ont été combinés à ceux de ma famille qui, malgré le flou qui régnait sur mon sujet de Doctorat, a tenté de comprendre mes délires d'informaticiens. Heureusement, Kenza était la pour canaliser ces crises de démences et apporter, sur mes travaux, l'avis d'un non initié aux Systèmes Multi-Agents.

Bien entendu, certains initiés ont encadré mon travail et orienté mes recherches. Je profite de ces lignes pour les remercier, en particulier : Olivier Boissier, l'équipe SMA de l'EMSE et les membres du laboratoire Hubert Curien. Je remercie également l'Ecole de Mines de Saint-Etienne pour l'infrastructure matérielle dont elle m'a fait bénéficier. Ce séjour dans ses locaux m'a donné l'occasion de me rapprocher de certains doctorants. Plus précisément, Ludivine, Guillaume et Rahee avec qui on a pu partager des idées sur nos travaux respectifs mais également de nombreux et bons moments.

Ces rencontres ont été possibles grâce au financement CIFRE établit avec la société E.C.A.I. Son dirigeant, Mr Vinzant, nous a été présenté par Jean-Claude Barthélémy, professeur au CHU de Saint-Etienne, qui m'a co-encadré durant mon stage de Master et apporté son expertise sur l'obésité. Je remercie ces deux personnes ainsi que les employés d'E.C.A.I et les médecins qui m'ont aidé au cours des ces trois années.

Enfin, je tiens à remercier tous mes amis qui tout du long de mon Doctorat ont su me changer les idées : JF pour les soirées passées avec un joystick et un bon jeu d'arcade, Stevens pour les pauses Snooker, Vincent pour son initiation au Japonais, Arnaud, Céline, Sylvain, Pierre, Jérémy et tous les autres, qui m'ont accompagné depuis le DEUG et ceux qui m'accompagnent désormais.... à la plage.

Ma liste n'est pas exhaustive, j'oublie certainement des personnes. Je tiens seulement à ce qu'elles sachent que si leur nom n'est pas cité sur le papier, il est présent dans mes souvenirs. S'ils me lisent, je leur passe le salut et attend avec impatience le moment de les revoir !!!

Résumé

Ce travail de recherche est consacré à la formalisation et à la réalisation d'un processus de diagnostic pluridisciplinaire. La particularité d'un tel diagnostic résulte du fait qu'il nécessite de nombreux spécialistes, chacun ayant des connaissances sur leur domaine. Le problème principal réside dans les interconnexions entre les domaines. Ces interconnexions peuvent ou non être connues et influer sur le diagnostic. Dans ce manuscrit, nous proposons de réaliser un diagnostic pluridisciplinaire à l'aide d'un système multi-agents. Les agents élaborent un diagnostic local à un domaine puis, fusionnent leurs diagnostics afin d'obtenir le diagnostic pluridisciplinaire. Dans ce but, nous proposons un cadre d'argumentation et une méthode de fusion des diagnostics. Ensemble, ces deux propositions forment le modèle ANDi.

Table des matières

Table des figures

Introduction

Ce manuscrit présente le travail réalisé en collaboration avec l'entreprise E.C.A.I. et le C.H.U. de Saint-Étienne. La problématique à laquelle nous nous sommes intéressé au cours de la thèse est celle du diagnostic pluridisciplinaires. C'est-à-dire, l'élaboration d'un diagnostic par de multiples acteurs issus de diverses disciplines. Dans la suite, nous argumentons le fait qu'un diagnostic pluridisciplinaires est une approche plus pertinente qu'un diagnostic établi sur un unique domaine. Puis, nous décrivons le contexte scientifique dans lequel se situe notre travail de recherche avant d'en énoncer les objectifs et les contributions. Enfin, nous détaillons la structure de ce manuscrit.

Motivations

Généralement, la résolution d'un problème nécessite un travail de groupe. Ce travail de groupe permet d'obtenir une richesse, mais également l'exhaustivité, dans l'identification et l'analyse des causes du problème. Pour cela, il est recommandé que l'équipe soit pluridisciplinaire, i.e. composée d'experts aux compétences variées. Cette recommandation est d'autant plus valable que de nombreux problèmes sont eux-même pluridisciplinaires et requièrent de multiples compétences (en médecine, par exemple, le diagnostic différentiel d'un patient fait intervenir plusieurs spécialistes). Ainsi, il est fréquent que des experts, possédant une expertise personnelle sur une ou plusieurs disciplines relatives au problème traité, coopèrent à l'élaboration d'une solution. Cependant, il n'est pas toujours évident de faire collaborer ensemble de nombreux experts. Cela demande à la fois, une forte implication des experts dans le processus d'analyse et un partage de leur connaissance.

Dans le cadre de notre travail, nous nous sommes intéressés à la réalisation d'outils de diagnostic. Ces outils sont destinés à être appliqués sur des problèmes multi-acteurs : la télé-consultation de l'obésité et le laminoir circulaire. Ces problèmes requièrent la participation d'experts issus de plusieurs disciplines pour être diagnostiqués. Notre outil permet de tenir

compte de l'expertise de chacun de ces experts dans le but de créer un argumentaire et de justifier ainsi la validité et la pertinence du diagnostic réalisé. La collaboration étant omniprésente, nous nous intéressons à reproduire de telles interactions à l'aide d'un système multi-agents et de l'argumentation.

Contexte scientifique

Le diagnostic du système étant un processus multi-acteurs, notre étude se situe dans le cadre des systèmes multi-agents. Nous nous intéressons plus précisément à des agents cognitifs et autonomes.

Chaque agent du système est un programme informatique qui représente un expert humain utile pour l'élaboration du diagnostic. Dans le cadre du manuscrit, nous considérons des agents cognitifs i.e. des agents capables de manipuler des connaissances déduites des expériences passées des experts humains. Ainsi, les agents possèdent des connaissances variées et doivent coopérer ensemble pour partager ces connaissances. Les agents doivent donc être capables de coopérer avec d'autres agents du système multi-agents. Le diagnostic devant être prouvé par les connaissances des agents et le raisonnement qu'ils ont tenu, nous nous intéressons également à l'argumentation. L'argumentation permet l'élaboration d'un argumentaire justifiant la décision finale des agents. Cet argumentaire repose sur les raisonnements personnels et collectifs des agents.

En résumé, notre travail se focalise sur la réalisation d'un diagnostic à l'aide d'un raisonnement individuel et collectif d'un ensemble d'agents cognitifs et autonomes.

Objectifs et contributions

Cette thèse contribue à l'étude et l'élaboration d'un modèle de diagnostic réparti. Contrairement aux modèles actuels qui modélisent le fonctionnement du système, nous cherchons à réaliser ce diagnostic à l'aide d'un ensemble de connaissances réparti entre les différents agents du système. Ces agents ont alors pour but de collaborer ensemble pour échanger leurs connaissances individuelles et réaliser ainsi, le diagnostic du système. De plus, nous souhaitons que le diagnostic réalisé soit compréhensible et puisse aider les utilisateurs du système.

Nous défendons la thèse selon laquelle le diagnostic peut être modélisé par un processus argumentatif au travers duquel les agents échangent et justifient leurs opinions. À ce titre, nous proposons un cadre d'argumentation. Ce cadre d'argumentation est le coeur du raisonnement des agents et sert de base aux outils de diagnostic que nous avons élaborés. L'élaboration d'un tel processus argumentatif se décompose en trois sous-objectifs :

1. *Cadre d'argumentation probabiliste*. La définition d'un cadre formel dans le but que les agents puissent tenir un raisonnement à partir de leur connaissances et des opinions des autres agents. Les arguments énoncés sont évalués par l'agent grâce à ses connaissances individuelles.
2. *Diagnostic par argumentation probabiliste.* L'élaboration de diagnostics basés sur le cadre d'argumentation probabiliste. Dans un premier temps, les agents ayant des connaissances sur une discipline commune dialoguent entre eux pour élaborer des *diagnostics locaux*. Ces diagnostics locaux donnent une tendance sur l'état du système (soutenue par un argumentaire) permettant d'améliorer et de justifier le *diagnostic global* du système.
3. *Outil de diagnostic*. La définition d'un outil de diagnostic pluridisciplinaires permettant de réaliser le diagnostic et mettre en oeuvre le diagnostic par argumentation probabiliste.

Organisation du manuscrit

Ce manuscrit est organisé en trois parties. La première partie dresse un état de l'art des diverses techniques utilisées dans la réalisation d'un diagnostic. La seconde partie décrit notre modèle. La troisième partie aborde sa mise en oeuvre et son illustration au travers de deux applications.

Partie I : État de l'art. Nous nous intéressons, dans cette partie, à la réalisation d'un diagnostic réparti entre plusieurs acteurs. Dans cette perspective, nous commençons par introduire les techniques majoritairement employées dans la réalisation d'un diagnostic. Puis, nous expliquons comment ces techniques ont été adaptées dans le cas d'un diagnostic réparti.

Diagnostic : approches existantes. Dans ce chapitre, nous introduisons les principales techniques utilisées pour élaborer un diagnostic. Parmi ces techniques, nous distinguons celles basées sur les symptômes et celles basées sur des modèles. La première catégorie s'intéresse à déterminer l'origine d'un dysfonctionnement en considérant ses effets sur le système (i.e. les symptômes). Le raisonnement mené pour élaborer le diagnostic est proche du raisonnement humain. La seconde catégorie, quant à elle, s'intéresse à modéliser le fonctionnement attendu du système que l'on souhaite diagnostiquer. Les modèles réalisés sont ensuite utilisés pour comparer le fonctionnement courant du système à son fonctionnement attendu.

Néanmoins, ces deux catégories possèdent un inconvénient commun : elles sont inadaptées aux systèmes actuels souvent répartis ou de grande ampleur. C'est pour cette raison que nous nous intéressons à l'élaboration d'un diagnostic de manière répartie.

Diagnostic réparti. Dans ce chapitre, nous montrons comment réaliser un diagnostic de manière répartie et en quoi il est intéressant de distribuer la tâche du diagnostic. L'intérêt principal du diagnostic réparti réside dans la modélisation du système en tant qu'un ensemble d'acteurs ou d'entités. Il est ainsi possible de réaliser le diagnostic du système en s'appuyant sur les déductions réalisées localement par les acteurs ou au sein des entités. Les relations entre les acteurs ou les entités sont modélisées à l'aide de mécanismes de collaboration.

Dans notre cas, la réalisation d'un modèle décrivant le fonctionnement attendu du système n'est pas envisageable. En effet, les experts ne possèdent pas les compétences techniques pour le faire. Cependant, ils possèdent des connaissances sur le procédé ou une discipline relative à ce procédé. Par conséquent, il est envisageable d'élaborer un diagnostic réparti basé sur les connaissances et les symptômes. L'argumentation permettant la collaboration des acteurs/entités et le raisonnement sur des connaissances, nous nous intéressons plus particulièrement à ce domaine.

Argumentation. Dans ce chapitre, nous focalisons notre attention sur l'argumentation et introduisons les caractéristiques nécessaires à la création d'un cadre d'argumentation ou d'une logique argumentative.

Ayant étudié les techniques utilisées pour le diagnostic multi-acteurs, nous sommes en mesure de réaliser une synthèse des travaux existant et de proposer notre propre modèle.

Partie II : Modèle. Cette partie décrit le modèle ANDi qui permet à une communauté d'agents de collaborer ensemble pour l'élaboration d'un diagnostic. Le raisonnement des agents repose sur un cadre d'argumentation qui permettant aux arguments d'être évalués selon la base de connaissances des agents et l'expérience des experts humains qu'ils représentent.

Cadre d'argumentation probabiliste. Dans ce chapitre, nous définissons la logique argumentative qui constitue le modèle de raisonnement des agents. Cette logique argumentative permet de gérer les contradictions entre arguments dont la force a été attribuée par un agent en fonction de ses connaissances et de l'expérience de l'expert humain qu'il représente.

Dans le chapitre suivant, nous montrons comment nous utilisons un tel cadre d'argumentation pour la réalisation d'un diagnostic local.

Système multi-agents pour le diagnostic local. Dans ce chapitre, nous présentons comment les agents interagissent et élaborent un diagnostic local. Ce diagnostic est réalisé par un ensemble d'agents représentant des experts humains spécialisés dans une discipline commune.

Ces diagnostics locaux servent ensuite à la réalisation du diagnostic global du système. Ce diagnostic prend en compte les relations entre les diverses disciplines.

Diagnostic global. Dans ce chapitre, nous détaillons la réalisation du diagnostic global. Ce diagnostic considère à la fois les relations entre les disciplines et les diagnostics locaux pour déterminer l'origine du problème.

Le modèle ANDi ayant été décrit, nous sommes en mesure de synthétiser notre contribution et d'en proposer un outil de diagnostic applicable sur divers problèmes multi-acteurs.

Partie III : Application. Cette partie décrit deux outils de diagnostic qui s'appuient sur le modèle ANDi.

Télé-consultation de l'obésité. Ce chapitre présente une application : la télé-consultation de l'obésité. Cet outil a pour but d'établir un diagnostic en utilisant les connaissances de plusieurs disciplines médicales.

Laminoir circulaire. Ce chapitre présente l'outil de diagnostic réalisé pour déterminer l'origine de défauts sur des pièces usinées sur un laminoir circulaire. Cet autre outil de diagnostic permet de montrer la généricité du modèle et sa réutilisabilité sur un système réel.

Implémentation. Ce chapitre présente les tâches communes réalisées dans la mise en oeuvre du modèle dans les applications du laminoir circulaire et de la télé-consultation de l'obésité.

Première partie

État de l'art

Chapitre 1

Diagnostic : approches pour le diagnostic

Sommaire

1.1 Introduction

Le terme diagnostic[1] désigne par étymologie, l'acquisition de connaissances à travers un ensemble de signes observables. De façon plus générale, il s'agit de raisonner sur les effets d'une défaillance ou d'un problème pour en déterminer les causes. Cette définition est appliquée dans divers domaines comme la médecine ou l'industrie.

Cependant, les approches respectivement utilisées dans ces domaines pour raisonner et diagnostiquer les pathologies ou les pannes divergent. En effet, en médecine, le clinicien établit

[1]D'après le dictionnaire Larousse.

son diagnostic à partir des symptômes observés ou recueillis auprès du patient lors de la consultation ou d'examens complémentaires. Le clinicien a donc connaissance de chaque signe observable. Ce type de diagnostic est dit *'hors ligne'*. A l'instar de la médecine, les méthodes appliquées en diagnostic de pannes sont majoritairement des méthodes dites *'en ligne'*, i.e. appliquées durant le fonctionnement du système ou de la machine. Par conséquent, la perception des signes observables (ici, les alarmes prévisibles ou non) est sujette au dysfonctionnement du système observé : des pertes ou des perturbations sont à prévoir.

L'Intelligence Artificielle s'intéresse aux problématiques liées au diagnostic : la détection, la classification et l'explication des problèmes . Cet intérêt a permis la réduction des coûts de maintenance et une meilleure fiabilité des systèmes de product1.2ion mais, a également fourni de nouveaux outils de diagnostic tant en médecine que dans l'industrie.

Dans ce chapitre, nous présentons brièvement les principales approches déployées en Intelligence Artificielle. Ces approches seront regroupées selon deux catégories : les approches basées sur les symptômes (cf. section 1.2) et celles basées sur des modèles (cf. section 1.3). Les critères auxquels doivent répondre totalement ou partiellement les outils de diagnostic sont introduits en section 1.4 [REF VENKA]. Dans la section 1.5, nous reprenons ces critères pour comparer et mettre en avant les avantages et les limites des approches basées sur les symptômes et basées sur des modèles.

1.2 Approches basées sur les symptômes

Relativement proches du diagnostic médical, les approches basées sur les symptômes s'intéressent à . L'idée principale est de construire le diagnostic en simulant le raisonnement humain. Les systèmes experts ont été introduits dans ce but. De tels systèmes utilisent des connaissances, acquises auprès d'experts humains, représentées sous forme de règles pour raisonner et expliquer la ou les causes d'un dysfonctionnement. D'autres travaux ont envisagé que ces connaissances puissent être apprises automatiquement et l'expérience simulée par l'introduction de techniques d'apprentissage automatique.

Dans cette section, nous proposons de passer en revue les principales techniques d'Intelligence Artificielle utilisées ou développées pour le diagnostic. Nous commençons par introduire les systèmes experts (cf. section 1.2.1) qui, par leur fonctionnement, sont les plus proches du raisonnement humain. Puis on présentera, dans la section 1.2.2, les quatre sous-familles de l'apprentissage automatique qui permettent d'acquérir ou de modéliser connaissances et expérience, mais également de construire un diagnostic fondé sur les symptômes.

1.2.1 Systèmes Experts

Les systèmes experts [53][44][68] ont été développés dans le milieu des années 60 afin de répondre à des questions par le biais d'un raisonnement par déduction. Ils sont principalement utilisés comme outils d'aide à la décision ou de conseils. Un système expert est composé de trois parties distinctes : une base de faits, une base de règles et un moteur d'inférence.

- La base de faits décrit le problème. Ces faits dénotent des paramètres ou des données du problème et, peuvent être utilisés durant le raisonnement mené par le système expert.
- La base de règles contient les connaissances spécifiques à la résolution du problème. Ces règles dites expertes (ou règles d'inférence) sont issues d'un ou plusieurs experts humains, d'où le nom des systèmes experts. Généralement, les systèmes experts reposent sur des mécanismes de logique formelle et utilisent des règles définies de la façon suivante :

  $$SI\ \mathsf{P}\ \text{est vrai et}\ \mathsf{P} \Rightarrow \mathsf{Q}\ \ ALORS\ \mathsf{Q}\ \text{est vrai.}$$

  Mises ensemble, la base de faits et la base de règles permettent d'inférer de nouvelles conclusions et d'aboutir à une ou plusieurs solutions. Ce travail de déduction est réalisé par le moteur d'inférence.
- Le moteur d'inférence [25][65], contrairement à la base de règles est indépendant du problème. Ce dernier permet à partir de la base de faits et de la base de règles de construire un raisonnement logique et déductif ayant pour but de décrire la ou les solutions trouvées par un ensemble de règles et de faits certifiés.

Les systèmes experts possèdent de nombreux avantages dont leur rapidité et leur robustesse. En effet, le raisonnement mené par un système expert est peu complexe et autorise la manipulation d'un grand nombre de faits et de règles expertes **[TODO]**. Par ailleurs, leur mise en place est facilitée par des outils de génération de systèmes experts tels que JESS [54] ou CLIPS [24]. Néanmoins, le principal avantage des systèmes experts réside dans les règles d'inférence. Étant le produit des experts humains, elles offrent des explications aisément interprétables et compréhensibles par les utilisateurs.

C'est pour ces raisons que les systèmes experts furent largement utilisés au milieu des années 70. Le lecteur intéressé pourra se référer à [68] et [64] qui survolent un certain nombre de ces systèmes experts. Le plus connu reste certainement Mycin [111][112][19] qui manipulait une base composée d'environ 500 règles pour diagnostiquer les maladies infectieuses du sang et recommander un antibiotique approprié au traitement de l'infection détectée. Actuellement, les systèmes experts ont laissé la place à des outils et des techniques plus performantes (cf. section

1.3), adaptés aux systèmes industriels d'aujourd'hui. On peut, néanmoins, citer le projet Sachem [41][45] actuellement utilisé par le groupe Usinor pour la surveillance et le contrôle de 6 hauts-fourneaux.

La perte d'intérêt ressentie par les systèmes experts est due à leur inadaptation aux données numériques et aux entraînées par la suppression de règles. Bien que les règles expertes soient le point fort des systèmes experts, elles sont également leur point faible. Elles permettent difficilement de s'adapter à l'évolution du problème ou du système traité sans une remise en cause de l'intégralité de la base de règles. De plus, ces règles manquent de généricité ; étant spécifiques à un problème donné, elles ne sont pas ou peu réutilisables. Enfin, l'expert humain avant de pouvoir modéliser efficacement le problème a besoin d'obtenir l'expérience nécessaire pour le faire. Par conséquent, le système expert ne peut pas être opérationnel tant que l'expert humain n'a pas acquis suffisamment d'expérience.

Dans la section suivante, nous présentons certaines méthodes de classification permettant, à partir de données recueillies lors de l'apparition du problème, de proposer statistiquement des règles d'inférence (cf. sections 1.2.2.1 et 1.2.2.2). Il existe cependant d'autres méthodes de classification pouvant se substituer aux systèmes expert et s'employer dans le contexte du diagnostic. Contrairement aux méthodes citées précédemment, ces dernières n'emploient pas de règles d'inférences. Nous introduisons ces méthodes dans les sections 1.2.2.3 et 1.2.2.4.

1.2.2 Méthodes de classification

Selon sa définition, un diagnostic désigne l'action d'identifier un dysfonctionnement à partir de ses symptômes. En général, cette tâche est attribuée à des experts qui grâce à leur expérience étiquettent le dysfonctionnement. Il est cependant évident qu'un diagnostic peut être défini à l'aide d'un classifieur, i.e. un modèle statistique. La classification, un sous-domaine de l'apprentissage automatique, s'intéresse à la découverte de tels modèles statistiques. On s'intéresse dans les prochains paragraphes aux principales catégories de classifieurs utilisées pour le diagnostic.

1.2.2.1 Raisonnement par déductions statistiques

Le raisonnement est assimilé à un mécanisme d'inférence. Dans le cas du raisonnement par déduction, ce mécanisme s'appuie sur un ensemble de règles causales dont la structure est équivalente à la structure des règles d'inférences précédemment introduites (cf. section 1.2.1). Il s'agit ici, d'acquérir les connaissances autrement que par l'intermédiaire d'un expert et d'utiliser les connaissances ainsi acquises pour raisonner.

Le but des techniques d'apprentissage présentées ci-dessous est de découvrir des relations causales au sein de données décrivant les cas précédemment observés. Ces relations causales sont ensuite exploitées pour attribuer, si nécessaire, un dysfonctionnement aux futurs cas.

- *Les arbres de décisions* : Les arbres de décisions sont le résultat de techniques d'apprentissage supervisées [93][92]. Ils permettent de répartir une population d'individus en groupes homogènes, selon un ensemble de variables discriminantes. Ils permettent également de prédire, en fonction de ces mêmes variables discriminantes le groupe d'appartenance d'un nouvel individu.
- *Les règles d'associations* : Les méthodes apprenant des règles d'associations [116][117] sont principalement utilisées pour la découverte de corrélations fréquentes, a priori inconnues, entre les éléments décrivant un ensemble de cas. Ces corrélations sont formulées sous forme de règles d'implication de la forme $X \Rightarrow Y$ et permettent d'associer une conclusion Y à un ensemble d'éléments X appelé prémisse. Les règles d'associations permettent de prédire les conclusions plausibles à partir d'un ensemble de prémisses.
- *Les réseaux Bayésiens* : introduits par Pearl [84][21], les réseaux Bayésiens représentent, sous forme de graphes orientés acycliques, les dépendances causales entre les variables définissant un ensemble d'individus. Contrairement aux arbres de décisions, les réseaux Bayésiens sont composés de relations causales probabilisées [12]. De ce fait, l'observation d'une ou plusieurs variables influe sur le résultat et tend à fournir une explication plus représentative de l'observation.

L'avantage des techniques présentées ci-dessus est d'être relativement proches du raisonnement humain. Les règles ainsi apprises et leur représentation sont compréhensibles et lisibles par l'utilisateur et peuvent servir, après validation par un expert humain, comme base pour un système d'aide à la décision. Dans le cadre d'un diagnostic hors ligne (cf. section 1.1), les réseaux Bayésiens sont néanmoins plus utilisés [98][104][63] car la représentation qu'ils proposent possède l'avantage d'attribuer une probabilité aux règles en fonction des faits observés (i.e. les données ou paramètres relatifs au problème).

1.2.2.2 Programmation logique inductive

La programmation logique inductive (PLI) [78][80] est à l'intersection entre l'apprentissage inductif et la programmation logique. La PLI reprend les techniques d'induction d'hypothèses à partir d'observations ou d'exemples pour définir de nouvelles connaissances. La PLI repousse les limites de ces techniques en utilisant une logique du premier ordre. Plus formellement, la PLI consiste, à partir des connaissances d'un domaine B et d'un ensemble

d'observations E (regroupés en E^+ et E^- représentant respectivement les exemples positifs et négatifs); à induire une hypothèse H répondant aux contraintes suivantes :

$B \wedge E^- \not\models \square$: *satisfiabilité* a priori[2] (1)

$B \wedge H \wedge E^- \not\models \square$: *satisfiabilité* a posteriori (2)

$B \not\models E^+$: *nécessité* a priori (3)

$B \wedge H \models E^+$: *condition* a posteriori (4)

Les contraintes ci-dessus spécifient que les connaissances du domaine ne sont pas suffisantes pour dériver tous les exemples positifs mais qu'elles sont nécessaires à l'hypothèse à la fois pour dériver les cas positifs et discriminer les cas négatifs.

La construction de l'hypothèse H (cf. Figure 1.2.1)**[REF AGRAVAL SHRIKANT]**, vérifiant les contraintes 2 et 4, repose sur l'espace des versions. Elle consiste à naviguer entre l'hypothèse la plus spécifique et l'hypothèse la plus générale permettant de couvrir E^+. Pour cela, la PLI se base sur le paradigme d' « inverting resolution » qui consiste à spécifier ou généraliser des règles à partir de E^+.

On observe au niveau des systèmes de PLI, deux approches de résolution : 'specific to general' et 'general to specific'. Dans la première catégorie, on retrouve des systèmes comme *Golem* [79] ou *Progol* [81] qui généralisent une hypothèse en partant des exemples et des connaissances du domaine. La seconde catégorie quant à elle, applique la méthode inverse et

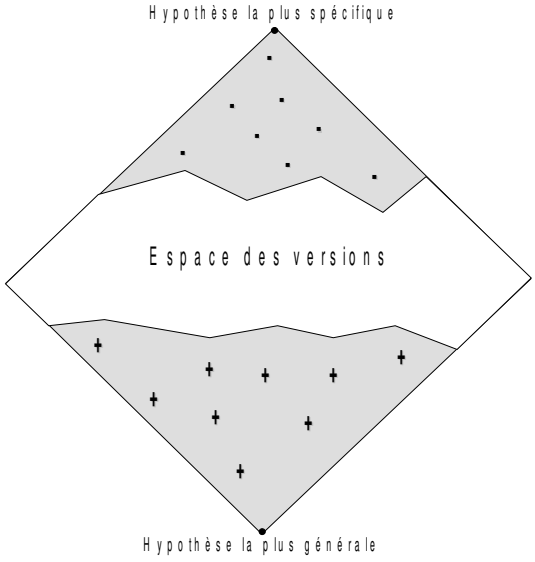


Figure 1.2.1 - Concept de l'apprentissage par induction

[2]Les symboles suivants $\wedge$, $\models$ et $\square$ représentent respectivement le *et logique*, la *preuve logique* et la *contradiction*

propose de spécifier une hypothèse jusqu'à la satisfaction des contraintes 2 et 4. Elle regroupe des systèmes tels que *Foil* [94] ou *Claudien* [28].

La PLI est généralement appliquée dans des domaines médicaux, comme la cardiologie [121] ou la pharmacologie [18][32] où les données sont fortement liées à leur représentation [40][39]. Par affiliation à *Prolog*, elle est également appropriée au traitement syntaxique et sémantique de langages. La PLI a l'avantage de produire des programmes logiques fortement liés au domaine traité même en l'absence d'exemples négatifs. Ces programmes, similaires à des programmes *Prolog* possèdent l'avantage d'être aisément traduits en règles logiques plus compréhensibles.

1.2.2.3 Raisonnement à base cas

Initialement fondé sur les travaux de Schank [106][107], le raisonnement à base cas [1] ou Case Based Reasoning (CBR) est une méthode d'apprentissage et de résolution. Contrairement aux approches présentées précédemment qui utilisent les connaissances propres au domaine, CBR s'appuie également sur des connaissances plus spécifiques : des cas. Pour résoudre de nouveaux problèmes, CBR les compare à des problèmes similaires déjà résolus. Les problèmes résolus alimentent ensuite la base de cas augmentant ainsi l'expérience du CBR et permettant de copier et/ou d'adapter la solution proposée à de nouveaux problèmes. Cette approche tend à copier le comportement humain mis en présence d'une situation inconnue.

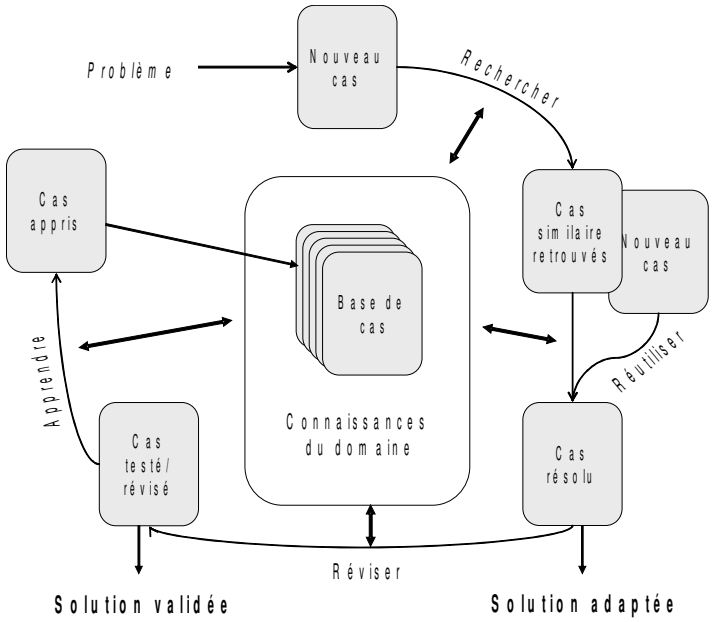


Figure 1.2.2 - Processus du CBR

Comme le montre la Figure 1.2.2, le CBR est un processus cyclique dont les deux phases principales consistent i) à résoudre un problème et ii) à acquérir de l'expérience de ce dernier. De manière détaillée, ce cycle peut être décrit par les quatre sous tâches suivantes :

- *Rechercher les cas similaires* : L'identification du problème débute par la recherche des cas similaires dans la base de cas. Cette recherche comprend la récupération de l'ensemble des candidats plausibles présents dans la mémoire du CBR mais aussi l'identification parmi ces candidats des cas correspondants le mieux au problème actuel. Cette sélection se fait à l'aide d'heuristiques ou d'algorithmes tel que l'algorithme des plus proches voisins [26].

- *Réutiliser les cas et les adapter* : Bien que les cas sélectionnés à l'étape précédente soient proches du problème actuel, ils sont rarement identiques. Cela implique qu'il faut adapter les solutions existantes, i.e. détecter les similitudes réutilisables.

- *Réviser* : Après sa génération par le CBR, la solution du problème est testée. Cette étape, généralement externe au CBR peut-être réalisée soit par un expert, soit par un logiciel de simulation. Si l'évaluation est concluante, l'expérience (la définition du problème et sa solution) est ajoutée à la base de cas du CBR. Sinon le CBR détermine les erreurs de raisonnement en utilisant les connaissances spécifiques au domaine et enrichit ainsi ses connaissances.

- *Apprendre* : La dernière phase du processus sert à déterminer quelles informations vont être sauvegardées dans la mémoire du CBR.

L'efficacité d'un CBR dépend grandement de sa mémoire [1], i.e. la représentation interne des cas. Cela s'explique par l'importance des phases de recherche et d'apprentissage du processus.

Par ses similitudes avec le raisonnement humain, le CBR est souvent appliqué à des problèmes requérant une certaine expérience [115] comme le droit [9], le diagnostic [89] ou encore l'explication d'anomalies [66]. Le principal inconvénient de cette approche reste l'acquisition et la représentation des cas, i.e. la conceptualisation de l'expérience.

1.2.2.4 Réseaux de neurones

Les réseaux de neurones schématisent le fonctionnement du cerveau humain à l'aide de concepts mathématiques simples. D'un point de vu biologique, un neurone (cf. Figure 1.2.3) est une cellules nerveuse qui transmet à l'aide de signaux électriques une information. Plus précisément, un neurone reçoit en entrée les signaux provenant des neurones auxquels il est connecté par des dendrites, et émet en sortie une information synthétique par son axone. De façon similaire, les neurones artificiels (cf. Figure 1.2.4) ou neurones formels [74] sont connectés entre eux par des liaisons pondérées unidirectionnelles formant ainsi un réseau appelé réseau de neurones.

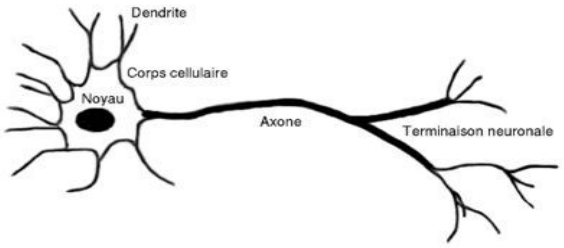


Figure 1.2.3 - Neurone

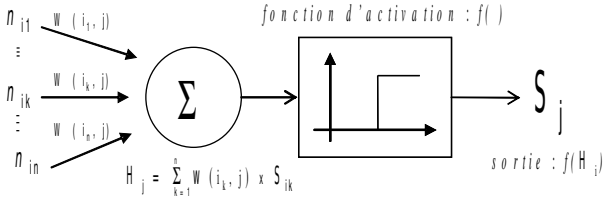


Figure 1.2.4 - Neurone formel

Initialement introduit par Rosenblatt [101] sous le nom de perceptron, les réseaux de neurones désormais multi-couches [52] permettent d'affecter une classe à un vecteur de données numériques. En général, un réseau de neurones est composé d'une succession de couches prenant chacune leurs entrées sur les sorties de la couche précédente (cf. Figure 1.2.5). Dans ces couches résident des neurones formels dont le but est de traiter les informations provenant des neurones de la couche précédente pour les propager à la couche suivante. Ce traitement consiste, pour un neurone *n* de la couche j, à multiplier les données de sortie $S_{i1},...,S_{in}$ des neurones n_{i1}, ..., n_{in} de la couche i par le poids des liaisons qui les relient (cf. Figure 1.2.4 et

1.2.5). Le résultat de ce calcul est ensuite utilisé comme paramètre d'une fonction, dite d'activation qui attribue la valeur de sortie du neurone *n*. Les données sont ainsi propagées entre la couche d'entrée, servant à la lecture des données à la couche de sortie qui lui affecte une classe. Les couches intermédiaires sont appelées couches cachées.

Développer un réseau de neurones se résume à définir son architecture (le nombre de couches et de neurones) et à pondérer ses liaisons. Il est cependant difficile de décider a priori de ces pondérations voire impossible. Par conséquent, il est courant d'associer au réseau de neurones un algorithme d'apprentissage pour modifier le poids des liaisons et minimiser les erreurs de classification [50][42].

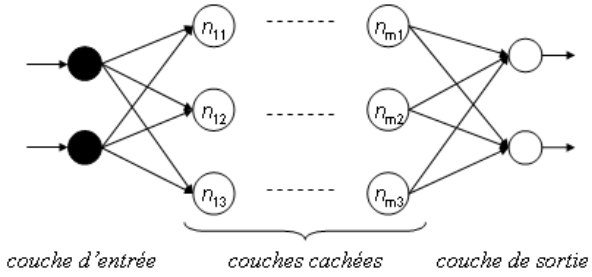


Figure 1.2.5 - Réseaux de neurones

Grâce à leur capacité de classification et de généralisation, les réseaux de neurones sont généralement utilisés en reconnaissance des formes, analyse du signal ou encore en diagnostic de panne [67][97][124]. En effet, associés à des méthodes telles que les ondelettes, les réseaux de neurones sont capables de traiter des problèmes complexes et d'associer analyse de signal et classification. Cependant, l'efficacité des réseaux de neurones dépend fortement de leur architecture, i.e. le type du réseau (Perceptron, ART, Réseaux de Hopfield) et le nombre de neurones.

1.2.3 Bilan

La classification s'emploie à trouver des corrélations au sein d'ensembles de données représentatives au problème traité. Les résultats des méthodes de classification sont par conséquent fortement dépendantes des données utilisées. Ainsi, même après acquisition d'un ensemble de données conséquent, il est possible que certaines typologies ne soient pas ou peu représentées. On introduit dès lors un biais amenant à des erreurs de classification.

Bien que facilitant l'acquisition de données, la classification possède les mêmes inconvénients que les systèmes experts : i) une phase d'acquisition de données obligatoire et ii) une phase d'extraction de connaissances dont le résultat dépend fortement des cas décrits par ces données.

Pour éliminer ce biais, il faut faire abstraction des données. Le problème ne sera alors plus résolu par expérience mais à l'aide d'un modèle décrivant le fonctionnement usuel du système.

1.3 Approches basées sur des modèles

Les approches basées sur les modèles s'intéressent à la représentation formelle ou schématique du système à étudier. Les données utilisées pour l'acquisition d'expérience, humaine ou statistique, ne sont alors plus nécessaires pour définir les problèmes et leurs explications. En effet, les causes du dysfonctionnement sont identifiées en comparant le comportement courant du système à son comportement attendu, estimé à la fois sur le modèle et les observations récupérées durant le fonctionnement du système **[TODO REF]**.

Dans cette section, nous introduisons les modèles quantitatifs (cf. 1.3.1) et qualitatifs (cf. 1.3.2) permettant respectivement la représentation mathématique et schématique du système ainsi que leur utilisation pour le diagnostic '*en ligne*'.

1.3.1 Modèles quantitatifs

Les modèles quantitatifs sont destinés principalement à la détection et l'isolation de pannes dans les systèmes dynamiques. On considère le système comme étant un ensemble de tâches séquentielles ou parallèles s'enchaînant en continue jusqu'à l'arrêt total du système observé. L'exécution des tâches engendre ou requiert des prises de mesures.

Dans le cas des modèles quantitatifs, l'estimation du comportement attendu consiste à prédire les valeurs en sortie d'une tâche à partir d'une série de mesures. Pour cela, on introduit des fonctions mathématiques qui définissent le fonctionnement habituel du système : les filtres de Kalman [4][58][57]. Un filtre de Kalman est un filtre récursif qui estime efficacement l'état d'un système dynamique à partir d'une série de mesures incomplètes ou bruitées. D'autres méthodes d'approximation sont également populaires parmi lesquelles : les équations de parité [43], les moindres carrés [69] ou l'approximation assistée par des observateurs [37].

Le comportement attendu, défini par les valeurs ainsi prédites est comparé au comportement courant (représenté par les mesures réelles) pour déterminer les inconsistances et identifier ainsi la tâche à l'origine du dysfonctionnement.

1.3.2 Modèles qualitatifs

Contrairement aux modèles quantitatifs qui décrivent le système à l'aide de fonctions mathématiques complexes, .

1.3.2.1 Réseaux de Petri

Un réseau de Petri [87] est un modèle mathématique adapté à la représentation de systèmes à événements discrets, i.e. travaillant sur des variables discrètes. Ces systèmes sont alors modélisés par un graphe bipartie orienté qui contient des arcs connectant entre eux des noeuds places et des noeuds transitions.

Formellement, un réseau de Petri est un quintuplet PN = $\langle P, T, F, W, M_0 \rangle$ où :

- P est l'ensemble des places,
- T est un ensemble de transition avec $P \cap T = \varnothing$ et $P \cup T \neq \varnothing$,
- $F \subseteq (P \times T) \cup (T \times P)$ est l'ensemble des arcs,
- $W : F \to \aleph^+$ est une fonction de poids et
- $M_O : P \to \aleph$ est le marquage initial.

Généralement, une place représente une ressource et une transition représente une action ou un événement. Les ressources consommées (resp. produites), dont la quantité est spécifiée par la fonction de poids W, sont décrites par les arcs appartenant à $(P \times T)$ (resp. appartenant à $(T \times P)$). Les places précédant (resp. succédant) une transaction sont appelées places d'entrées (resp. places de sortie).

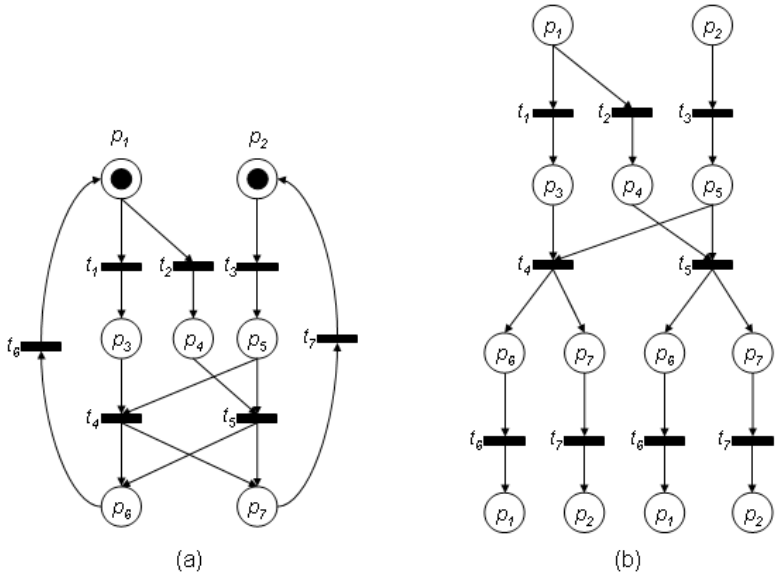


Figure 1.3.1 - Un réseau de Petri (a) et son dépliage (b)

Initialement, les ressources (également appelées jetons) sont réparties sur les places du réseau de Petri selon le marquage initial M_0 (cf. Figure 1.3.1a). Cette distribution évolue en fonction des transitions déclenchées lors de l'exécution. Une transition peut être déclenchée (on dit alors que la transition est déclenchable) uniquement si ses places d'entrées disposent des ressources nécessaires à son exécution. Autrement dit, le nombre de jeton de chaque place $p_{entrée}$

reliée à la transition t doit être supérieur au poids $W(p_{entrée}, t)$. Par ailleurs, le déclenchement de la transition t produit de nouvelles ressources distribuées à ses places de sortie p_{sortie} selon les poids $W(t, p_{sortie})$.

Dans un réseau de Petri, plusieurs transitions peuvent, si elles sont déclenchables, être exécutées simultanément. On peut, par conséquent modéliser des comportements concurrents. Le dépliage [62][51] (cf. Figure 1.3.1b), utilisé pour le diagnostic par réseau de Petri, s'intéresse à déterminer le marquage du réseau à la suite d'une séquence d'événements, i.e. le comportement observé. La Figure 1.3.1b est le dépliage du réseau de Pétri représenté dans la Figure 1.3.1a, ce dernier a été construit par suppression des boucles présentes dans le réseau de Pétri. Les comportements pouvant être concurrents, il peut exister plusieurs explications possibles pour un même ensemble d'observations. Néanmoins, on peut réduire ces possibilités en utilisant des réseaux de Petri stochastiques [87][23] et en connaissant l'ordre d'exécution des transitions.

1.3.2.2 Automates

Un automate est un modèle de comportement (cf. Figure 1.3.2) composé d'un ensemble d'états et de transitions qui relient ces états entre eux. Contrairement aux réseaux de Petri, ou les transitions schématisent des actions en terme de consommation et de production de ressources, les transitions d'un automate représentent les événements déclenchant le passage d'un état à un autre du système.

Par définition, un automate est un quintuplet $A = \langle S, \Sigma, \delta, s_0, F \rangle$ où :

- $S \neq \varnothing$ est l'ensemble des états possibles du système,
- Σ est l'ensemble des événements qui peuvent se produire dans le système,
- $\delta : S \times \Sigma \to 2^S$ est la fonction de transition,
- $s_0 \in S$ est l'état initial du système et
- $F \subseteq S$ est l'ensemble des états finaux du système.

Il est communément admis qu'une séquence d'événements $\sigma_0, \ldots, \sigma_n \in \Sigma$ et d'états $q_0, \ldots, q_{n+1}$ telle que $q_{i+1} \in \delta(q_i, \sigma_i)$ permet d'identifier l'évolution d'un système entre l'état q_0 et l'état q_{n+1}. Le diagnostic de systèmes, représentés à l'aide d'automates, s'intéresse à la découverte de telles séquences. Et plus particulièrement à identifier le ou les états potentiellement emprunté par le système selon l'ensemble d'événements observés.

Généralement, seulement certains événements sont observés. Dans ce cas, le diagnostic revient à rechercher toutes les séquences de l'automate, i.e. du modèle dans lesquelles apparaissent ces événements. L'automate ainsi construit, appelé automate diagnostic [46][86], représente les comportements potentiels du système.

Certaines approches tentent d'éliminer les séquences les moins probables de l'automate diagnostic. Dans [60], par exemple, le système est modélisé par un automate probabiliste appelé graphe de dépendances. À chaque état q_i de cet automate est associé un poids p_i représentant la probabilité que q_i soit à l'origine du dysfonctionnement. Un poids p_{ji} est également associé à la transition reliant q_i à q_j et spécifie la probabilité que q_j soit mis en échec par q_i. Enfin, les événements observables sont définis comme étant des alarmes émises par les éléments du système (i.e. les états de l'automate) affectés par le dysfonctionnement. Diagnostiquer le système consiste à trouver le sous-ensemble d'éléments $E \subseteq S$ expliquant les alarmes et maximisant la probabilité suivante : $P(E) = \sum_{\forall i : q_i \in E} (p_i + \sum_{\forall j : q_j \in E,\, j \neq i} p_{ji} \times p_i)$.

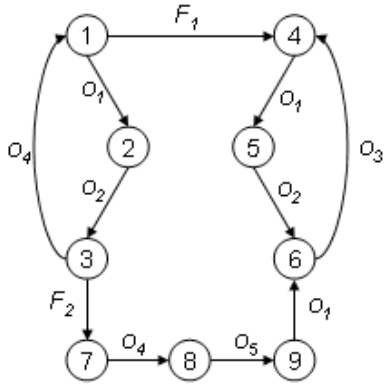


Figure 1.3.3 - Exemple de modèle

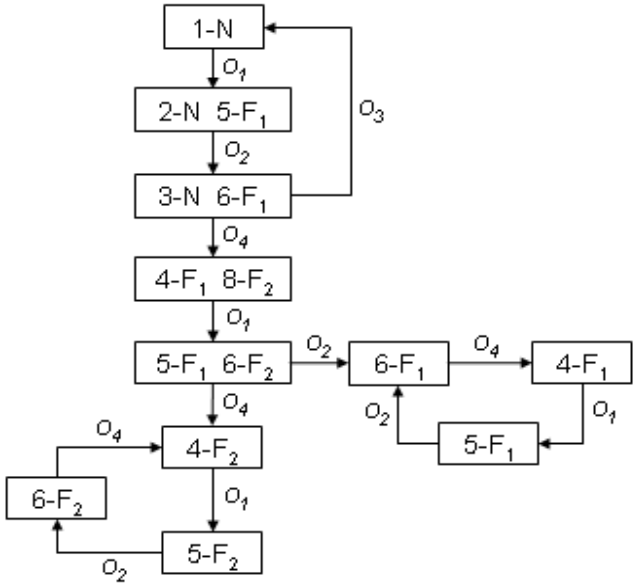


Figure 1.3.2 - Diagnostiqueur du modèle de la figure 1.3.2

D'autres approches [105][125] supposent que les événements non observés sont à l'origine des dysfonctionnements et qu'il est possible grâce au modèle de définir un autre type d'automate de diagnostic : un diagnostiqueur. Le diagnostiqueur (cf. Figure 1.3.2) est un automate dont les transitions sont étiquetées uniquement par les événements observables et dont les états sont définis comme un ensemble de couples état-panne. Un label particulier *N* est toutefois associé aux états du modèle ne subissant aucune panne. L'utilisation d'un diagnostiqueur est relativement simple. Il suffit de parcourir le diagnostiqueur en fonction des observations reçues. L'état du diagnostiqueur sur lequel aboutit le parcours indique le ou les états possibles du système ainsi que les pannes ayant pu se produire.

La construction du diagnostiqueur consiste à rendre le modèle déterministe après avoir propagé les pannes au niveau des états. Dans l'exemple (cf. Figure 1.3.3), les pannes F_1 et F_2 sont considérées comme des événements non observables au même titre que o_5. Ainsi, les transitions contenant un événement non observable comme $1 \xrightarrow{F_1} 4 \xrightarrow{o_1} 5$ sont remplacées au sein du diagnostiqueur par des transitions définies comme $(1-N) \xrightarrow{o_1} (2-N\,5-F_1)$ (l'état 2 étant également accessible par o_1).

Malgré la simplicité des automates, leur utilisation est souvent restreinte par des problèmes de passage à l'échelle. En effet, la taille des modèles dans de nombreux cas réels (par exemple, les réseaux de télécommunication) est très importante et demande d'énormes capacités de traitement surtout dans le cadre d'un diagnostic '*en ligne*'.

1.3.3 Bilan

Les approches de diagnostic basées sur des modèles possèdent l'énorme avantage de pouvoir être définies sans l'apport d'un ensemble de données propre au système. Par conséquent, de telles approches sont opérationnelles immédiatement et éliminent le biais introduit par les données. Néanmoins, ces approches sous-entendent que l'on connaisse a priori le système observé et son comportement. Elles sont destinées à être appliquées sur des problèmes fonctionnels ce qui est le cas, par exemple, des applications industrielles.

L'inconvénient majeur de ces approches est certainement leur manque d'adaptabilité. Une fois le modèle défini, il sert uniquement à la détection de dysfonctionnements dans le système pour lequel il a été modélisé. Chaque reconfiguration d'un système entraîne donc des modifications au niveau de son modèle.

1.4 Évaluation des systèmes de diagnostic

Dans cette section, nous détaillons les principaux critères d'évaluation et des systèmes de diagnostic proposés par [REF Venka] en expliquant leurs enjeux. Ces critères, bien que fortement liés aux problèmes et au type de diagnostic déployé sont révélateurs des performances des systèmes de diagnostic. Ces critères sont organisés en trois groupes : 1) les caractéristiques intrinsèques du système, 2) l'évolution du système et 3) l'aide à l'utilisateur.

1. Caractéristiques intrinsèques du système : nous rassemblons dans ce premier groupe les caractéristiques fortement liées aux performances des systèmes de diagnostics ainsi qu'à leur bon fonctionnement.
   - ***Identification de pannes multiples*** : Il arrive que plusieurs dysfonctionnements ayant des effets communs se déclenchent simultanément. Le système de diagnostic doit alors détecter et isoler chacun de ces dysfonctionnements malgré leur apparition conjointe.

- ***Rapidité du diagnostic*** : Dans le cadre d'un fonctionnement en ligne, le système doit être capable de détecter et diagnostiquer rapidement les dysfonctionnements afin d'éviter leur développement. Cependant, une détection rapide a des impacts sur les performances du système de diagnostic. Les systèmes répondant à cette caractéristique intègrent des mécanismes d'inférence ou de calcul rapides sensibles pour la plupart aux données bruitées et aux pannes récurrentes.
- ***Robustesse*** : La robustesse joue également sur les performances du système. Un système de diagnostic robuste se doit d'être peu sensible aux données bruitées et d'isoler correctement le ou les dysfonctionnements, demandant ainsi des traitements complexes.
- ***Scalabilité***[3] : Les problèmes de grande échelle requièrent des modèles complexes, des ensembles de données conséquents et des connaissances suffisantes pour la mise en place du système de diagnostic. Le réalisation de ces modèles (resp. le traitement des données) et leur exploitation influe également sur les performances.

2. Évolution du système : ce groupe rassemble deux caractéristiques fortement souhaitées pour assurer la maintenabilité et l'évolutivité de l'outil de diagnostic.
   - ***Adaptabilité :*** L'environnement lié au problème est amené à changer. Par exemple, les prescriptions du patient peuvent être modifiées ou le processus délocalisé. Le système de diagnostic doit donc prendre en compte ces changements pour établir son diagnostic.
   - ***Identification de nouvelles pannes*** : Il est préférable pour un système de diagnostic d'identifier les comportements anormaux même s'ils sont a priori inconnus, i.e. si aucune typologie ne lui est associée.
3. Aide à l'utilisateur : ce dernier groupe rassemble deux critères qui évaluent la capacité du système de diagnostic et facilitent l'interprétation du diagnostic par les utilisateurs.
   - ***Facilité d'explication*** : Mis à part la détection et l'identification du dysfonctionnement, les systèmes de diagnostic doivent également être capables d'expliquer les causes et les effets du dysfonctionnement. Dès lors, l'utilisateur a la possibilité d'interpréter ces explications et de les évaluer suivant son expérience personnelle.

[3] Capacité d'un système, ou de ses composants, à être utilisé sur des plates-formes de tailles très inférieures ou très supérieures, Wiktionnaire (http://fr.wiktionary.org).

- ***Estimation de l'erreur*** : L'estimation de l'erreur donne une information supplémentaire à l'utilisateur qui lui permet, en plus des explications fournies par le système de diagnostic, de juger de la pertinence du diagnostic.

Dans la section 1.5, nous nous proposons de comparer les approches précédemment introduites à l'aide des critères d'évaluation cités dans la présente section et de mettre en évidence les avantages et inconvénients de chacune d'elle.

1.5 Synthèse

Dans ce chapitre, nous avons brièvement présenté différentes méthodes couramment utilisées pour le diagnostic. Ces méthodes principalement issues de l'Intelligence Artificielle répondent en partie aux caractéristiques désirées d'un système de diagnostic (cf. section 1.4).

Les systèmes expert et les méthodes de classification offrent la possibilité de travailler à partir d'une expérience obtenue auprès d'un expert humain ou d'un logiciel. De telles méthodes sont appropriées en l'absence de connaissances fondamentales utiles à la définition d'un modèle quantitatif. Les modèles causals (tels que les réseaux Bayésiens ou les arbres de décisions) sont aussi efficaces dans de telles circonstances, mais proposent une représentation symbolique des dépendances fonctionnelles facilitant les explications et l'interprétation du diagnostic. Néanmoins, acquérir de l'expérience nécessite une phase d'apprentissage

pendant laquelle le système de diagnostic n'est pas opérationnel. Cet apprentissage est relatif aux situations rencontrées. L'expérience ainsi acquise ne représente que partiellement le fonctionnement réel du système rendant les approches basées sur les symptômes peu appropriées à la détection de nouvelles pannes.

Tableau 1 - Comparaison de diverses méthodes de diagnostic

	Système expert	Réseau Bayésien	Réseau de neurones	Automate
Identification de pannes multiples	✗	✓	✗	✓
Détection et diagnostic rapide	✓	*?*	✓	*?*
Robustesse	✓	✓	✓	✓
Scalabilité	✗	✗	✗	✗
Adaptabilité	✗	✗	✗	✗
Identification de nouvelles pannes	✗	✗	✓	✓
Facilité d'explication	✓	✓	✗	✓
Estimation l'erreur	✗	✓	✗	✗

Les approches basées sur les modèles n'affichent pas un tel inconvénient puisque le modèle définit le fonctionnement du système dans sa totalité. De telles approches offrent également de nombreux autres avantages. Les modèles quantitatifs approximent avec une grande précision le comportement du système. Quant aux modèles qualitatifs, ils permettent d'expliquer la propagation des dysfonctionnements, ce qui s'avère utile pour l'aide à la décision. Cependant, la définition de modèles quantitatifs ou qualitatifs requiert des connaissances approfondies sur le système mais également des compétences techniques.

Le Tableau 1 compare les méthodes représentatives des approches présentées précédemment selon les critères d'évaluation définis dans la section 1.4. Le symbole '✓' indique que la méthode spécifiée (colonne) satisfait le critère (ligne). La croix indique que le critère n'est pas satisfait. Enfin, le point d'interrogation indique que la satisfiabilité du critère varie selon le problème traité.

La méthode de diagnostic à intégrer dans le système de diagnostic dépend fortement du problème. Pour la surveillance et à la maintenance de systèmes industriels, il est préférable d'utiliser des méthodes permettant une détection rapide des pannes tels que les réseaux de neurones, les systèmes experts ou encore des modèles quantitatifs. Les méthodes de classification sont, quant à elles, recommandées pour l'explication de dysfonctionnements déjà apparus dans le système observé.

Néanmoins, toutes ces approches possèdent un inconvénient commun : elles sont inadaptées aux systèmes actuels souvent répartis et/ou de grande ampleur [86]. Afin de répondre à cette limitation, nous allons introduire dans le chapitre suivant, la notion de diagnostic réparti.

Chapitre 2

Diagnostic réparti

Sommaire

2.1 Introduction

De nos jours, les systèmes de grande envergure ont tendance à être décentralisés. Souvent ces systèmes sont constitués de composants ou d'acteurs interdépendants dont le but est d'assurer le bon fonctionnement du système. Les dépendances entre composants peuvent être physiques ou dénoter un quelconque échange d'informations (par exemple, des échanges de données ou d'hypothèses).

Concevoir un système à l'aide de composants a de nombreux avantages dont le principal est l'évolution aisée du système (par le changement d'un composant). Les composants rendus génériques sont réutilisables dans divers contextes. Les acteurs, quant à eux, possèdent des connaissances sur le système ou le secteur d'activité (appelé domaine par la suite), ils sont capables de les manipuler, de les échanger et de se construire une opinion personnelle basée sur les opinions d'autres acteurs. Ainsi, utiliser une architecture répartie permet de simuler le mode de fonctionnement (resp. le mode de raisonnement) des composants (resp. des acteurs) tout en réduisant la complexité des calculs.

En effet, modéliser un système par ses composants et ses acteurs permet de définir des modèles simples et ainsi de réduire la complexité du diagnostic. Les approches décentralisées et distribuées s'intéressent respectivement à ces modélisations ; elles cherchent toutes deux à élaborer le diagnostic du système en fusionnant les diagnostics élaborés à partir des modèles propres aux composants (appelés diagnostic locaux). Les approches distribuées se différencient par la construction de diagnostics intermédiaires pouvant englober plusieurs composants.

Dans ce chapitre, nous commençons par présenter les architectures basées sur des approches décentralisées (cf. section 2.2) avant de nous intéresser aux architecture distribuées (cf. section 2.3) et aux architecture distribuées hiérarchiques (cf. section 2.4). Enfin, nous présentons en section 2.5, les moyens utilisés par ces architectures pour s'échanger des informations et fusionner les diagnostics locaux.

2.2 Architectures de diagnostic décentralisées

Les architectures décentralisées sont principalement utilisées pour réduire la complexité du diagnostic. Pour cela, le modèle du système observé Γ est décomposé en un ensemble de modèles Γ_1, ..., Γ_n décrivant chacun une partie de ce système. Des diagnostics partiels $\Delta_{\gamma 1}$,..., $\Delta_{\gamma n}$, également appelés diagnostics locaux, sont alors calculés grâce aux observations $\Theta=\{\theta_{\gamma_1},\ldots,\theta_{\gamma_n}\}$, aux modèles Γ_1, ..., Γ_n et aux comportements locaux $\|\gamma_1\|,\ldots,\|\gamma_n\|$ observés à la suite des observations Θ. Le diagnostic global du système Δ est alors calculé par un unique superviseur [86]. Ce diagnostic est le résultat de la fusion des diagnostics locaux (cf. Figure 2.2.1).

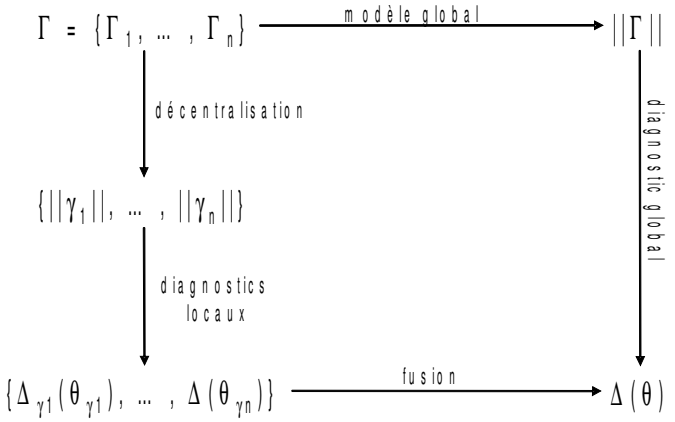


Figure 2.2.1 - Approche décentralisée

Dans la suite, nous présentons quatre approches de diagnostic dites décentralisées. Ces approches s'appuient sur des techniques de diagnostic, telles les diagnostiqueurs ou les réseaux de Pétri, introduites dans le Chapitre 1 et permettent de calculer le diagnostic du système Δ à partir des diagnostics partiels $\Delta_{\gamma 1}$,..., $\Delta_{\gamma n}$.

2.2.1 Approche à la Debouk et al.

L'approche proposée dans [27] considère un système comme étant un ensemble de sites en relation avec un coordinateur. Chacun des sites symbolise une entité physique et reçoit certaines observations sur l'environnement. Plusieurs de ces sites peuvent avoir besoin des mêmes observations et doivent donc être capables de communiquer celles dont ils disposent localement (par exemple, la valeur d'un capteur).

Dans [27], on suppose que le système est modélisé par un modèle global Γ dont les transitions sont étiquetées par les observations. Ainsi, il est possible de construire pour chaque site un modèle partiel cohérent avec sa vision du monde, i.e. les observations qu'il reçoit. Le modèle Γ est mis à disposition du coordinateur. Les modèles partiels sont, quant à eux, mis à disposition des sites qui les utilisent pour calculer un diagnostic local basé sur les observations qu'ils reçoivent. Ce diagnostic est généralement établi par l'utilisation d'un diagnostiqueur (cf. chapitre 1).

Grâce à un protocole de communication, les informations et les résultats des différents sites sont communiqués au coordinateur qui les utilise pour l'inférence (cf. Figure 2.2.2). Plus précisément, le coordinateur déduit la séquence du diagnostiqueur que le système a empruntée, en s'appuyant sur les visions partielles de chacun des sites. Les incohérences dénotent alors la présence d'un dysfonctionnement.

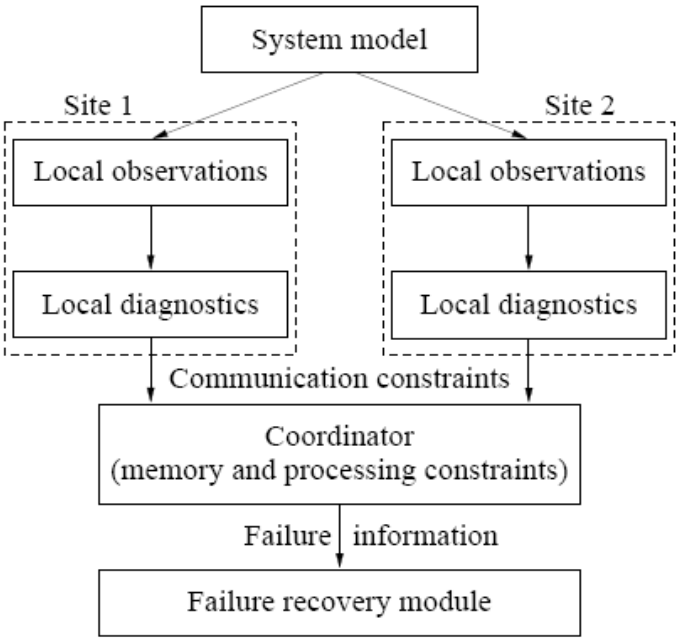


Figure 2.2.2 - Architecture de l'approche

Malgré la répartition des calculs auprès des sites, l'approche nécessite la définition d'un modèle global. Ainsi, toute modification au niveau du système observé aura un impact sur les modèles du coordinateur et des sites. L'approche est destinée à des systèmes dont la structure est statique afin d'éviter les modifications répétées du modèle. Dans les sections suivantes, nous présentons des approches destinées à des systèmes dynamiques, i.e. des systèmes dans lesquels les sites peuvent être reconfigurés par l'ajout ou le retrait d'un ou de plusieurs composants (ou acteurs).

2.2.2 Approche à la Aghasaryan et al.

Dans [2], le système, distribué selon plusieurs sites, est observé par des managers. Chaque site possède son propre manager qui reçoit les alarmes et possède quelques connaissances sur leur dépendance. Les alarmes sont considérées comme des observations partiellement ordonnées notifiant la présence d'un dysfonctionnement au manager. L'ordre est défini en fonction de la perception de ces alarmes par les managers et peux donc différer d'un site à l'autre.

Ces hypothèses permettent de décrire explicitement la propagation des dysfonctionnements à l'aide de motifs. Plus précisément, la propagation représente les dépendances entre les alarmes et les dysfonctionnements et peut être modélisée par des réseaux de Petri (cf. chapitre 1), appelés '*réseaux de dysfonctionnement*'. Les réseaux de Petri ainsi

créés, sont constitués de noeuds représentant les alarmes et de transitions représentant les dépendances possibles entre deux alarmes.

La détection d'un dysfonctionnement se fait alors par l'identification de son motif. C'est à dire en parcourant les réseaux de Petri représentatifs des dysfonctionnements et en discriminant ceux qui n'expliquent pas les alarmes reçues. L'utilisation de réseaux stochastiques permet de réaliser cette tâche de manière plus robuste [2].

2.2.3 Diagnostic décentralisé par automates communicants

Similairement aux réseaux de Petri, les automates communicants permettent de modéliser les systèmes décentralisés aisément. Chaque partie du système étant représentée par un automate, les interactions entre les différentes parties sont reproduites par des envois de messages. Le mécanisme de fusion est alors basé sur ces interactions et peut être réalisé par reconstruction (cf. section Erreur : source de la référence non trouvée2.2.3.1) ou par synchronisation (cf. section 2.2.3.2Erreur : source de la référence non trouvée).

2.2.3.1 Approche par reconstruction

Dans [10], Baroni et al. s'intéressent au diagnostic de systèmes distribués de grande échelle au sein desquels se produisent des événements asynchrones. Baroni et al. présupposent que de tels systèmes sont cycliques et réactifs, i.e. évoluent suite à la perception d'événements observables. Ces événements observables peuvent être soient caractéristiques d'un dysfonctionnement, soient le résultat du fonctionnement courant du système (valeur d'un capteur, diffusion d'information, etc). Le but de leur approche est alors de déterminer le comportement du dit système grâce à ces événements, tout comportement non prédictible étant représentatif d'un dysfonctionnement.

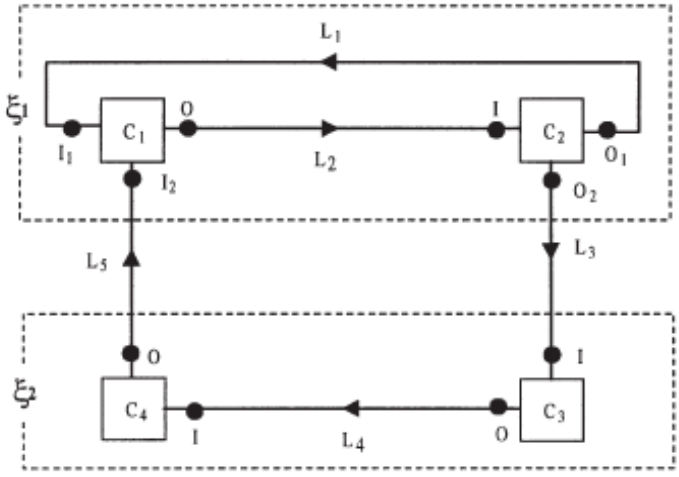


Figure 2.2.3 - Représentation de l'approche de Baroni

Par construction (cf. Figure 2.2.3), de tels systèmes sont constitués de composants (C_1, C_2, C_3 et C_4) distribués reliés entre eux par des liaisons physiques (L_1, L_2 et L_4) appelées liens. Selon la topologie du système, ces composants sont regroupés en clusters (ξ_1 et ξ_2) également reliés par des liaisons physiques appelées interfaces (L_3 et L_5). La modélisation proposée dans [10] définit un composant comme un automate communicant (cf. Figure 2.2.4 automates M_1, M_2, M_3 et M_4). Les noeuds de cet automate correspondent à des réactions du composant aux événements, eux mêmes représentés par une transition précisant le ou les événements déclenchés suite aux traitements réalisés dans le noeud (cf. Figure 2.2.4 transition T_{ij}).

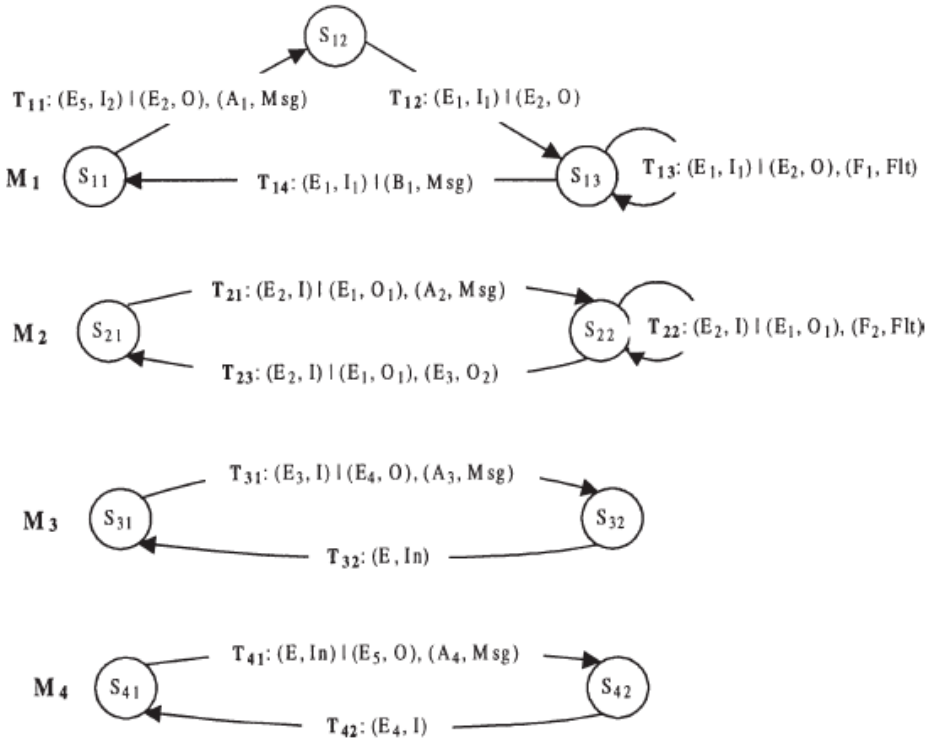


Figure 2.2.4 - Automates des composants C_1 à C_4

Diagnostiquer le système revient à reconstruire son historique (cf. Figure 2.2.5), i.e. déterminer les réactions de chaque composant en fonction des événements observables générés par le système. Les processus de diagnostic se déroulent en trois étapes :

- *L'interprétation* : L'interprétation consiste, pour chaque cluster (ξ_1 et ξ_2) du système, à générer un historique expliquant les événements observés. Ces historiques sont, en quelque sorte, les modèles des différents clusters et sont construits à l'aide des modèles des composants (cf. Figure 2.2.4).
- *La reconstruction* : La phase de reconstruction s'intéresse à fusionner les historiques des clusters reliés par une interface. La Figure 2.2.5 présente l'historique du système issu de

la fusion des historiques des clusters ξ_1 et ξ_2 reliés entre eux par les interfaces L_3 et L_5. Le résultat de la fusion représente l'historique du système dans son intégralité.

- *La génération du diagnostic* : La génération du diagnostic cherche à identifier les comportements relatifs à un dysfonctionnement compris dans l'historique du système. Ainsi, chaque séquence de l'historique (cf. chapitre 1, section 1.3.2.2) comprenant au moins une transition imprévue fournit une explication au dysfonctionnement.

L'avantage d'une telle approche réside dans la répartition des calculs selon les clusters. En effet, diagnostiquer un système sur un modèle global requiert beaucoup de ressources. En traitant les clusters séparément, on peut alors réduire l'espace de recherche ainsi que le temps de calcul. De plus, un diagnostic partiel peut être envisagé, considérant alors qu'un sous-ensemble de clusters.

D'autres stratégies de fusion ont été également proposées dans [86]. Elles sont basées sur la dépendance des diagnostics locaux. Le principe est de réduire l'espace de recherche en fusionnant dans un premier temps, les composants interagissant le plus. Cette méthode est similaire à celle introduite dans cette section.

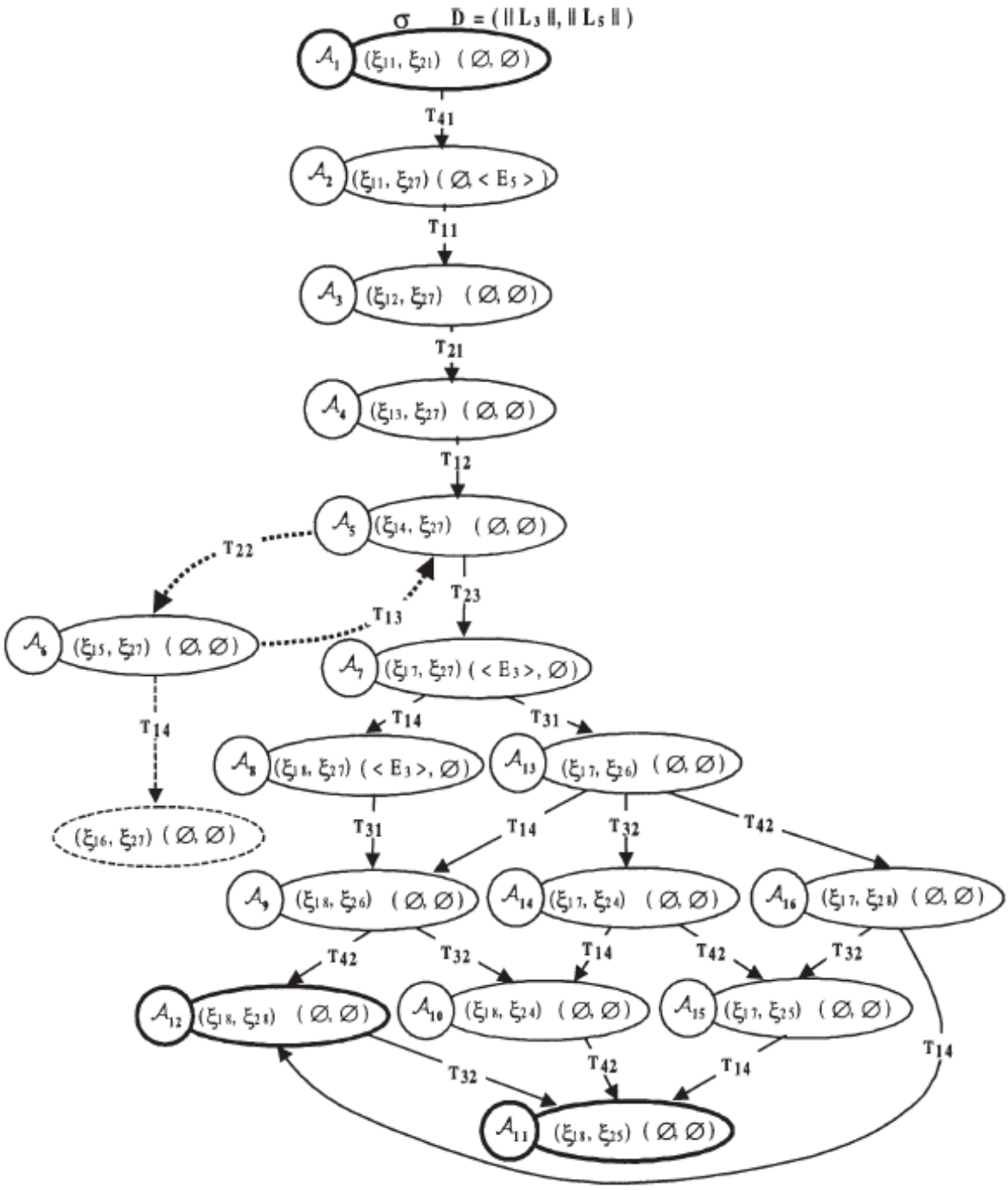


Figure 2.2.5 - Diagnostic généré par la fusion des historiques des clusters ξ1 et ξ2

2.2.3.2 Approche par synchronisation

Le formalisme proposé dans [102] est très semblable à celui proposé par Baroni et al. [10]. Le système est également modélisé à l'aide d'automates communicants décrivant le comportement des composants qui le constituent. Additionné à ce modèle de comportement, un modèle structurel précise comment les messages sont propagés entre les composants du système.

Un composant est modélisé par un sextuplet $C = \langle Q, \Sigma, \delta, q_0, V, A \rangle$ où :

- Q est l'ensemble des états et $q_0 \in Q$ est l'état initial,
- Σ est l'ensemble des événements reçus Σ_r, émis $\Sigma_e$ou internes Σ_{endo},
- δ est un ensemble de transitions,
- V est un ensemble de variables et $A(V)$ les actions correspondantes possibles.

Cette formalisation permet d'obtenir le modèle global du système par synchronisation des automates communicants, i.e. des modèles des composants. L'opération de synchronisation consiste à fusionner ensemble, un ou plusieurs automates. Considérons les modèles de deux composants $C_1 = \langle Q_1, \Sigma_1, \delta_1, q_{0\ 1}, V_1, A_1 \rangle$ et $C_2 = \langle Q_2, \Sigma_2, \delta_2, q_{0\ 2}, V_2, A_2 \rangle$, l'automate résultant de la synchronisation de C_1 et C_2 est $S = \langle Q_1 \times Q_2, \Sigma = \Sigma_1 \cup \Sigma_2, \delta, q_0, V_1 \cup V_2, A_1 \cup A_2 \rangle$ avec $\delta : Q_1 \times Q_2 \times \Sigma_r \rightarrow Q_1 \times Q_2 \times (\Sigma_e)^*$. Un exemple de synchronisation est donné dans la Figure 2.2.6.

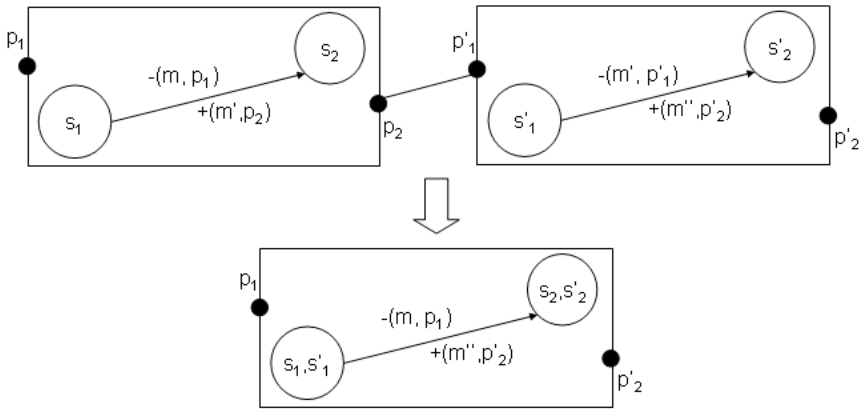


Figure 2.2.6 - Synchronisation d'automates

Dans la Figure 2.2.6, les deux composants sont reliés par le lien (p_2, p'_1). La réception du message m sur le port p_1 déclenche le passage de C_1 (à gauche) à l'état s_2 ainsi que l'émission d'un message m' sur le port d'écoute p_2. De façon similaire, C_2 (à droite) passe de l'état s'_1 à l'état s'_2 à la réception de m' et émet un message m'' sur son port de sortie p'_2. La synchronisation de C_1 et C_2 est décrite par l'automate $C_1 \otimes C_2$ (en bas) passant de (s_1,s'_1) à (s_2, s'_2) à la réception de m sur son port d'entrée p_1 et émettant le message m'' sur le port p'_2.

Le modèle global, une fois construit, autorise la création d'un diagnostiqueur (cf chapitre 1). Cependant, il est communément admis que la taille du modèle est limitative. Afin de limiter la complexité du diagnostic, [46] propose de synchroniser les automates sur des plages de temps. Les observations désormais associées à une plage de temps [t_{i-1}, t_i] sont représentées également par un automate Obs^i. Le diagnostic local du composant j est alors défini comme $\Delta^i_j = C_j \otimes Obs^i_j$[4]. Ainsi, le diagnostic du système sur la plage de temps [t_{i-1}, t_i] peut être écrit $\Delta^i = \Delta^i_1 \otimes ... \otimes \Delta^i_n$. Les explications du dysfonctionnement correspondent alors aux trajectoires générées à partir de l'ensemble d'automates $\epsilon_{\Delta^i} = (\Delta^1, ..., \Delta^i)$ [48][47].

2.3 Architectures de diagnostic distribuées

Les architectures distribuées reprennent le même principe que les architectures décentralisées. Elles cherchent à définir un diagnostic global à partir d'un ensemble de modèles (resp. de connaissances) décrivant les différents composants (resp. acteurs). Cependant, le calcul du diagnostic global n'est plus centralisé, mais réalisé par plusieurs superviseurs ayant chacun la charge d'un ou plusieurs composants (ou acteurs).

Les architectures décrites ci-après sont caractérisées par leur manière de construire le diagnostic global du système. Ainsi, les superviseurs peuvent reconstruire le diagnostic global en fonction des liens entre les composants du système observé comme le propose Fabre et Ross

[4] $\otimes$ représente l'opérateur de composition.

(cf. section 1.3.1Erreur : source de la référence non trouvée et 1.3.2) ou, par composition asynchrone (cf. section 1.3.3).

2.3.1 Approche à la Fabre et al.

L'approche proposée dans [31] se situe dans la même optique que celle proposée par Baroni [10] et Debouk [27] (cf. section Erreur : source de la référence non trouvée2.2.3.1). Cependant, l'état du système n'est pas estimé à partir d'un historique réalisé par un unique superviseur. Au contraire, Fabre déduit cet état d'une multitude de vues locales fournies par des superviseurs locaux en charge d'une ou plusieurs parties du système.

Chaque sous-système est modélisé à l'aide d'un automate pondéré (cf. Chapitre 1). Dès lors, estimer l'état du système consiste à déterminer les transitions qui ont été déclenchées dans les divers sous-systèmes connaissant une séquence d'événements. Les trajectoires, également appelées séquences (cf. chapitre 1), compatibles avec les événements observés représentent alors les comportements possibles du système. Elles sont construites par les superviseurs locaux qui i) se communiquent les trajectoires compatibles avec les k premiers événements, ii) ajoutent les transitions compatibles avec le $k+1^{ème}$ événement et iii) éliminent les trajectoires non satisfaisantes, c'est-à-dire les trajectoires ayant une probabilité faible.

2.3.2 Approche à la Roos et al.

L'approche introduite par Roos [99][100] s'intéresse également à distribuer les connaissances à des superviseurs locaux en fonction de la topologie du système. Pour cela, il définit un système comme un tuple $S = (C, M, Id, Sd, Ctx, Obs)$ où :

- C est un ensemble de composants possédant chacun un modèle M_c décrivant leurs pannes ($M = \{Mc \mid c \in C\}$),
- Id décrit les points de connexions, i.e. les liaisons physiques reliant les composants,
- $Sd = Str \cup Beh$ est la description structurelle et comportementale du système,
- Ctx est une spécification des valeurs extérieures au système, i.e. issues de son environnement,
- Obs est l'ensemble des valeurs observées par le système, i.e. diffusées par les points de connexions.

Ainsi, la représentation d'un sous-système $S_i = (C_i, M, Id, Sd_i, Ctx, Obs_i, In_i, Out_i)$ peut être attribuée à chaque superviseur, où C_i, Sd_i et Obs_i sont les données relatives au sous-système S_i. Les ensembles In_i et Out_i (inclus dans Str) sont respectivement les connexions d'entrée et de sortie des composants de S_i.

Diagnostiquer le système consiste à élaborer les diagnostics D_i de chaque sous-système S_i. L'élaboration de D_i repose sur la description Sd_i du sous-système S_i et tend à déterminer les dysfonctionnements des composants de S_i en utilisant à la fois leur modèle M, le contexte environnemental Ctx et le contexte local des composants, i.e. les valeurs de Obs circulant sur les points de connexions de In_i. Ce contexte local est important car il permet également d'évaluer le fonctionnement des sous-systèmes dont S_i dépend. En effet, la cause d'un dysfonctionnement dans le sous-système S_i peut-être dû à la propagation d'une panne. Dans un tel cas, le diagnostic D_i met en évidence les valeurs incohérentes ou entrant en conflit avec les déductions du superviseur, et permet de prévenir les sous-systèmes intéressés qui adapteront leurs diagnostics en conséquence. Au terme de ces échanges, le diagnostic du système est défini comme étant la synthèse des diagnostics locaux.

2.3.3 Approche par composition asynchrone

Dans la même logique que les deux précédentes approches, Benveniste propose dans [15] [16] d'assurer la modularité du système en utilisant des superviseurs locaux prenant à leur charge la supervision d'un composant dont le modèle est défini à l'aide d'un réseau de Petri (cf. chapitre 1). Ces composants étant interconnectés, certaines places (dites partagées) seront présentes dans différents réseaux de Petri.

L'arrivée d'un dysfonctionnement dans le système provoque l'arrêt du composant et l'émission d'une alarme. Cette alarme est ensuite propagée aux superviseurs en charge d'un composant relié au composant défectueux. Le but des superviseurs est d'expliquer les alarmes émises. Ils utilisent pour cela, le dépliage du réseau de Petri de leur composant (cf. chapitre 1). Néanmoins, les superviseurs ont seulement une vision locale du système. Ils doivent alors coopérer ensemble pour retrouver l'intégralité des comportements. Cette coopération consiste pour un superviseur s à informer ses voisins (les superviseurs partageant une place avec s) des transitions concernant les places partagées.

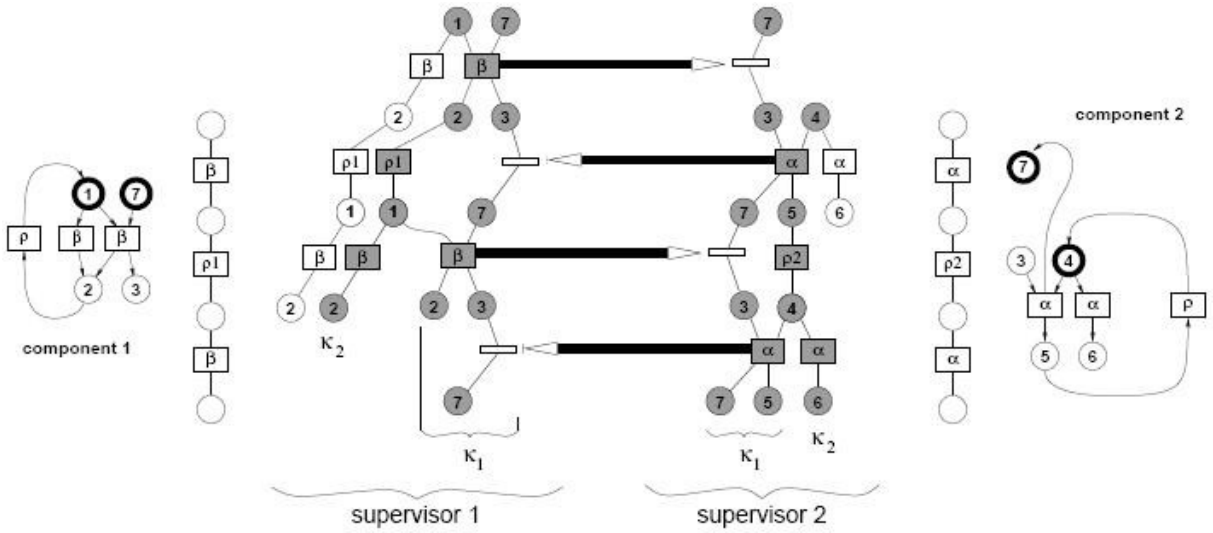


Figure 2.3.1 - Diagnostic distribué

Le processus de coopération est schématisé dans la Figure 2.3.1 dans laquelle sont représentés les dépliages des composants 1 et 2 réalisés respectivement par les superviseurs 1 et 2. Les comportements correspondants à la séquence d'alarmes $A = (\beta, \alpha, \rho, \rho, \alpha, \beta)$ sont représentés par les places et les transitions grisées. Les flèches signifient qu'il y a eu communication entre les superviseurs. Considérons le cas de l'alarme β, une transition est déclenchée dans le composant 1. Cette transition peut être déclenchée depuis la place 1 ou la place 7. La place 7 étant partagée avec le composant 2, le superviseur 1 informe le superviseur 2 de la possibilité que la place 3 soit atteinte. Dès lors, le superviseur 2 peut déterminer les places atteintes par le déclenchement de la transition α. Ce processus continue tant que les alarmes n'ont pas toutes été traitées. Les comportements K_1 et K_2 décrivent alors les deux comportements concurrents expliquant A.

2.4 Architectures de diagnostic distribuées hiérarchiques

Dans les architectures de diagnostic distribuées et hiérarchiques, les calculs sont également réalisés par plusieurs superviseurs. Néanmoins, ces superviseurs sont organisés en hiérarchie. Les superviseurs du niveau *n-1* sont surveillés par les superviseurs du niveau *n*. Les superviseurs appartenant au niveau le plus bas sont ceux gérant les composants. Le diagnostic global du système est réalisé par un superviseur du niveau *n* uniquement si la cause de la panne n'a pas été découverte par les superviseurs du niveau *n-1*. Nous présentons, dans les sections 1.4.1 à 1.4.3, trois approches fondées sur ce principe.

2.4.1 Approche à la Khanna

[61] introduit plusieurs types de superviseurs organisés de façon hiérarchique. Parmi ces superviseurs, certains surveillent les composants du système. Ces superviseurs, appelés '*Local Monitor*' (*LM*) détectent les dysfonctionnements en analysant les messages échangés par les composants et en les comparant avec une base de règles décrivant leur fonctionnement (cf. Chapitre 1).

Lorsqu'un dysfonctionnement est détecté, les *LM* regroupent les composants soupçonnés d'en être la cause. L'état de ces composants est ensuite testé par les *LM*. Dans certains cas, un *LM* peut soupçonner un composant n'appartenant pas au périmètre qu'il surveille d'être la cause du dysfonctionnement. Le *LM* entre alors en contact avec le *LM* intéressé par le biais d'un superviseur de plus haut niveau : un superviseur intermédiaire (*IM*). De façon similaire, le *IM* transmet ces demandes au *LM* en question si ce dernier est sous son contrôle. Auquel cas, il contacte un *IM* ou le superviseur en charge de la globalité du système.

Le diagnostic se fait récursivement jusqu'à ce que chaque composant supposé défectueux ait été testé par les *LM*. Le dysfonctionnement est alors expliqué par le raisonnement mené par les *LM* durant la procédure de test des composants.

2.4.2 COMMAS

L'architecture définie dans COMMAS [73][71] est également hiérarchique. Les composants sont surveillés par plusieurs superviseurs chargés d'identifier des anomalies dans leur fonctionnement. Ces superviseurs sont appelés '*Attribute Reasoning Agent*' (*ARA*). Contrairement à l'approche précédente [61], les *ARA* examinent les données générées par les composants à l'aide de techniques d'apprentissage telles que les réseaux de neurones, le clustering et les arbres de décisions (cf. Chapitre 1). Les classes définies par ces différents modèles désignent les dysfonctionnements présents dans le système.

Lorsqu'une anomalie est détectée au sein d'un composant, un vote est mené dans le but de déterminer le dysfonctionnement. Le résultat du vote est la classe majoritairement attribuée par les *ARA* surveillant ce composant. Un superviseur, nommé '*Cross Sensor Corroboration Agent*' (*CSCA*), utilise ensuite cette classe pour corroborer l'existence de ce dysfonctionnement en s'appuyant sur l'état des autres composants du système. Finalement, le diagnostic correspond au dysfonctionnement confirmé par le *CSCA*.

2.4.3 Approche à la Schroeder et al.

Dans [110] et [38], le système est considéré comme étant composé de plusieurs sous-systèmes. Chacun de ces sous-systèmes est surveillé par un superviseur ayant des connaissances détaillées sur le fonctionnement du sous-système qu'il surveille. Ces connaissances sont définies par un programme logique à partir duquel le comportement du sous-système peut être inféré (cf. chapitre 1 – PLI). Les superviseurs possèdent également une vue abstraite de leur voisinage et répondent à la définition proposé dans [119][108].

Ainsi, les superviseurs sont capables de détecter les comportements suspects de leur sous-système. [61] et [110] assument que la plupart des comportements suspects peuvent être découverts en local, i.e. sans coopération avec un autre superviseur. Néanmoins, les sous-systèmes sont interdépendants. Par conséquent, un comportement suspect peut être le résultat d'un dysfonctionnement propagé par un autre sous-système. Dans un tel cas, le superviseur entame une phase de coopération avec ses voisins en leur communiquant son diagnostic réalisé localement, aux sous-système voisins. Les diagnostics réalisés ensuite par les superviseurs voisins permettent de conclure que i) le sous-système marche correctement, ii) la panne est

intermittente ou iii) qu'un consensus est nécessaire. Ce consensus tente de résoudre les conflits mis en évidence par les superviseurs au travers de leurs diagnostics.

Le diagnostic du système est alors obtenu en retrouvant l'ensemble minimum des sous-systèmes affichant un comportement suspect, non explicable par la propagation d'un dysfonctionnement. C'est à dire, les sous-systèmes pour lesquels le consensus a abouti sur la confirmation du dysfonctionnement.

2.4.4 Bilan

Les approches décentralisées et distribuées partagent le même objectif : réduire la complexité du diagnostic global. Les approches décentralisées s'intéressent principalement à définir des techniques de fusion, permettant la construction du modèle du système, à partir d'un ensemble de modèles locaux. Ces architectures sont caractérisées par la présence d'un unique superviseur effectuant l'intégralité du raisonnement final.

Au contraire, dans les architectures distribuées, les raisonnements sont effectués localement aux sous-systèmes. Ces raisonnements sont réalisés par plusieurs superviseurs qui effectuent ensuite des recoupements sur les conclusions auxquelles ils ont abouti. En raison, de leurs capacités à interagir et à résoudre des problèmes, les agents [123][122] sont souvent utilisés pour tenir le rôle de superviseurs. Les messages précédemment échangés par les superviseurs sont enrichis et permettent des interactions plus complexes. La section suivante présente les mécanismes de collaboration pouvant dès lors être employés dans le cadre du diagnostic distribué.

2.5 Mécanismes de collaboration

Dans les architectures de diagnostic distribuées, on a pu constater que les superviseurs devaient collaborer pour diagnostiquer l'état général du système. Collaborer requiert des interactions entre les superviseurs présents dans le système. Ces interactions peuvent se faire de façon indirecte ou directe. Les interactions indirectes nécessitent un média de communication utilisé par les superviseurs pour s'échanger des informations ou leurs conclusions (cf. section 2.5.1). Les interactions directes, quant à elles, ont lieu de superviseur à superviseur. Les interactions directes sont du ressort de la communication et plus particulièrement du dialogue (cf. section Erreur : source de la référence non trouvée).

2.5.1 Collaboration par mémoire partagée

Dans le cadre d'une collaboration par mémoire partagée, les superviseurs interagissent au moyen d'un artefact commun appelé '*blackboard*' [82]. Ce *blackboard* sert à centraliser les

informations et les hypothèses proposées par les superviseurs. Il sert également de média de communication permettant aux superviseurs d'intervenir dans la résolution d'un problème.

Le scénario suivant, couramment utilisé pour expliquer le fonctionnement d'un *blackboard*, permet d'introduire simplement la mécanique sous-jacente au *blackboard* [82] :

> *« Plusieurs spécialistes se réunissent autour d'un tableau (blackboard) et travaillent ensemble dans le but de résoudre un problème. Pour cela, ils utilisent le tableau comme un espace de travail commun. Les spécifications du problème, une fois écrites au tableau, permettent aux spécialistes de l'étudier et d'apporter leur expertise dès qu'ils ont l'opportunité de le faire. Ainsi, les spécialistes développent une solution en s'appuyant sur les contributions précédemment écrites sur le tableau et en appliquant leur expertise. Au final, les contributions inscrites au tableau sont la résolution du problème. »*

Le *blackboard* permet donc, une interaction indirecte entre les superviseurs durant laquelle le problème est traité de manière incrémentale. Les superviseurs modifient le *blackboard* en ajoutant, supprimant, modifiant les hypothèses qui y sont inscrites apportant de nouvelles connaissances aux autres superviseurs qui pourront les utiliser et contribuer à leur tour, à création de nouvelles hypothèses. Au final, les superviseurs ont construit une solution partielle ou totale au problème.

2.5.2 Collaboration par échange de messages et dialogue

La collaboration n'implique pas forcément la centralisation des informations (cf section 2.5.1), elle peut également se faire en communiquant et échangeant ses opinions sans l'intermédiaire d'un *blackboard*. Pour cela, les superviseurs se fondent sur des protocoles de communication qui standardisent et structurent l'ensemble de leurs interactions. Dès lors, un dialogue est un ensemble d'interactions conformes à un protocole de communication spécifié a priori. Walton et Krabbe [120] ont identifié cinq types de dialogue (cf. Tableau 2) :

1. *La délibération* : La délibération consiste à déterminer la meilleure action à réaliser lorsqu'un dilemme ou un choix pratique est détecté. Par exemple, une délibération peut avoir lieu pour décider d'une heure de rendez-vous.
2. *L'enquête* : Une enquête définit une interaction entre plusieurs participants. Elle a pour but de répondre à une question ouverte dont la réponse est initialement inconnue des participants.
3. *La persuasion* : La persuasion est utilisée pour convaincre quelqu'un de la véracité d'une proposition. Bien qu'elle tente de résoudre ou de clarifier les conflits d'opinions,

elle est souvent utilisée en négociation pour influencer les participants ne partageant pas la même opinion.

4. *La négociation* : Contrairement à la persuasion, la négociation sert à résoudre les conflits d'intérêts afin d'obtenir une ressource désirée. Évidemment, si une phase de négociations est entamée, cela sous-entend qu'il existe des désaccords entre les participants. Négocier consiste alors à trouver un terrain d'entente satisfaisant au mieux les participants impliqués dans la négociation. Il est par conséquent courant d'avoir recours à la délibération et à la persuasion lors d'une négociation.

5. *La recherche d'informations* : L'acquisition d'une ou plusieurs informations se fait par le biais d'un dialogue de recherche d'informations. Une telle demande dénote le fait, qu'un participant souhaite obtenir une information concrète qui lui est initialement inconnue. Contrairement à l'enquête, il est sous-entendu qu'au moins un des participants a la réponse à cette demande d'informations.

Au niveau d'un système de diagnostic réparti (cf. section 2.3), ces différents types de dialogue sont importants. En effet, le diagnostic du système est le résultat de la mise en commun des opinions des superviseurs, i.e. diagnostics locaux. Du fait que les superviseurs ne possèdent pas les mêmes informations et les mêmes connaissances, certains conflits d'opinions peuvent apparaître. Les superviseurs doivent alors vérifier les hypothèses, se convaincre mutuellement et acquérir des informations complémentaires pour être en mesure de juger la pertinence de leurs diagnostics. Le but de leurs échanges est d'aboutir à un consensus dont les termes reflètent l'état global du système.

Tableau 2 - Classification des dialogues

Type de dialogue	Situation initiale	But du participant	But du dialogue
Délibération	*Dilemme ou choix pratique*	*Coordonner buts ou actions*	*Décider de la meilleure action*
Enquête	*Besoin de preuve*	*Trouver ou vérifier les évidences*	*Prouver les hypothèses*
Persuasion	*Conflit d'opinions*	*Persuader l'autre parti*	*Résoudre ou clarifier*
Négociation	*Conflit d'intérêts*	*Obtenir ce qu'il désire*	*Aboutir à un accord*
Recherche d'informations	*Besoin d'informations*	*Acquérir des informations*	*Échanger des informations*

Ces différents types de dialogue sont étroitement liés à l'argumentation. Plusieurs travaux [95][113] suggèrent l'utilisation de l'argumentation dans les processus de négociation afin de pouvoir connaître les circonstances d'un rejet ou d'une acceptation. Ces échanges d'arguments permettent d'orienter la négociation et facilitent la découverte d'un accord entre les participants. Par ailleurs, [5] et [96] proposent d'utiliser l'argumentation pour modéliser toutes les classes de dialogue introduites dans [120].

2.6 Synthèse

La conception d'un système de diagnostic au moyen d'une architecture répartie a de nombreux avantages. Ces avantages sont la conséquence d'une distribution des connaissances. En effet, cette distribution permet de gérer et de traiter des problèmes moins compliqués, mais également de concevoir des entités indépendantes facilitant ainsi l'évolution du système de diagnostic.

	Distribuée / Décentralisée	**Symptômes / Modèle**	**Collaboration**
2.2.1	Décentralisée	Modèle	Protocole
2.2.2	Décentralisée	Modèle	Réseaux de Petri
2.2.3.1	Décentralisée	Modèle	Automate
2.2.3.2	Décentralisée	Modèle	Automate
2.3.1	Distribuée	Modèle	Composition d'automates
2.3.2	Distribuée	Modèle	Composition d'automates
2.3.3	Distribuée	Modèle	Petri + Protocole
2.4.1	Distribuée hiérarchique	Modèle	Protocole
2.4.2	Distribuée hiérarchique	Symptômes	Vote
2.4.3	Distribuée hiérarchique	Symptômes	Protocole

Tableau 3 - Approches décentralisées et distribuées

Comme précisé précédemment, les architectures décentralisées (cf. Tableau 3, approches 2.2.1 à 2.2.3.2) s'intéressent plus particulièrement à la définition d'un modèle global, i.e. à la représentation symbolique du fonctionnement du système. De tels modèles nécessitent des compétences techniques approfondies qui ne sont pas toujours disponibles. Dans ce manuscrit, nous nous intéressons à des systèmes dont le fonctionnement exact est méconnu des experts. De

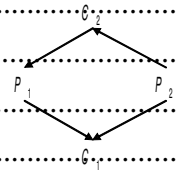
ce fait, la construction de modèles décrivant ces systèmes n'est pas réalisable. Nous ne pouvons pas appliquer à ces systèmes des approches basées sur des modèles. Par conséquent, la conception d'une approche basée sur les symptômes est préférable obligeant la réalisation d'une architecture distribuée (cf Tableau 3, approches 2.4.2 à 2.4.3). Afin de résoudre des problèmes pluridisciplinaires, nous avons opté pour une architecture distribuée hiérarchique dans laquelle des superviseurs sont chargés de surveiller les diverses disciplines (également appelées domaines). Un superviseur global gère le calcul du diagnostic en utilisant les diagnostics locaux précédemment réalisés par les superviseurs des disciplines (cf. section 2.2). Nous avons cependant besoin d'un mécanisme de collaboration pour faire interagir les superviseurs.

Notre choix s'est porté sur un *blackboard* qui permet de centraliser et d'échanger les informations entre les superviseurs. Néanmoins, nous dotons nos superviseurs de niveau 0 de la capacité d'enquêter et d'échanger des informations (cf. section 1.5.3). C'est pour cette raison qu'on se focalisera, dans la section suivante, sur l'argumentation qui comme nous l'avons expliqué permet de modéliser les différents types de dialogue spécifiés dans [120].

Chapitre 3

Argumentation

Sommaire



3.1 Introduction

L'argumentation [90][22] s'intéresse à construire interactivement une démonstration validant ou invalidant une proposition. Pour cela, plusieurs partis s'échangent leurs arguments, construits grâce à leurs connaissances, pour appuyer leurs conclusions ou mettre en doute celles

d'un autre parti. L'argumentation sert également à comprendre la raison d'un échec, tel est le cas en négociation [7][59][95][113] (cf. section 2.5.2). Toutefois, la négociation tend à argumenter sur la réalisation d'une tâche ou l'obtention d'une ressource. La décision finale dépend fortement des besoins et des concessions des partis prenant part à la négociation. C'est pour cette raison que nous nous intéresserons uniquement à l'argumentation en tant que moyen de démonstration interactive.

L'argumentation repose sur un échange d'arguments. Ces arguments ont été, dans un premier temps, définis par Dung [30] comme des entités abstraites uniquement reliées entre elles par une relation d'attaque. Cette relation est la base des travaux d'Amgoud [6][7] et Bench Capon [13] que nous introduisons dans la section 3.2 sous le nom de cadres d'argumentation. Désormais, une structure a été associée aux arguments. Cette structure a permis de définir des relations de contradiction reliant les arguments entre eux. Ces relations de contradiction permettent d'évaluer l'acceptabilité d'un argument, i.e. sa validité en présence d'un ou plusieurs autres arguments. Dans la section 3.3, nous présentons les différentes relations de contradiction existantes dans la littérature. Elles forment ce que nous appelons des logiques argumentatives. Dans de nombreux cas, il est nécessaire que les échanges respectent une convention, on parle alors de systèmes dialectiques (cf. section 3.4).

3.2 Cadres d'argumentation

Un cadre d'argumentation est une modélisation du raisonnement dans lequel, des arguments sont comparés et évalués à l'aide d'une relation dite d'attaque. Dung est le premier a avoir introduit cette notions dans [30]. Ses travaux ont été ensuite repris par Amgoud [6][7] et Bench Capon [13] qui proposent tous deux des extensions au modèle de Dung. Ces extensions s'intéressent à affiner la relation d'attaque initialement proposée par Dung en attribuant aux arguments une relation de force. Cette relation de force permet alors d'ordonner les arguments et de définir ainsi, de nouvelles relations entre les arguments.

Dans un premier temps, nous allons présenter les travaux de Dung (cf. section 3.2.1). Puis, chacune de ses deux extensions : le cadre d'argumentation à base de préférences (cf. section 3.2.2) et le cadre d'argumentation à base de valeurs (cf. section 3.2.3).

3.2.1 Cadre d'argumentation abstrait

Le but principal du cadre d'argumentation introduit par Dung [30] est de modéliser de façon générique l'argumentation humaine. Un argument est une entité abstraite dont le rôle est uniquement déterminé par la relation qu'il entretient avec d'autres arguments (cf. Définition 3.2.I).

Définition 3.2.I [Cadre d'argumentation]

> Un *cadre d'argumentation* est un couple $AF = \langle AR, attacks \rangle$ où :
> - AR est un ensemble fini d'arguments ;
> - $attacks$ une relation binaire sur AR ($attacks \subseteq AR \times AR$).
>
> Pour deux arguments A et $B \in AR$, la notation $attacks(A, B)$ signifie que A représente une attaque envers B.

Nous dirons que « A attaque B » ou encore que « B est attaqué par A » si $attacks(A, B)$. Par abus de langage, on dira qu'un ensemble d'arguments S attaque B (ou B est attaqué par S) si B est attaqué par au moins un argument de S, i.e. $attacks(S, B) \Leftrightarrow \exists A \in S, attacks(A, B)$.

Dans ce contexte, un ensemble d'arguments S est dit libre de conflit si les arguments de S ne s'attaquent. Par ailleurs, un argument est dit acceptable s'il peut être soutenu malgré les attaques de ses contre-arguments, i.e. s'il bénéficie de l'appui d'un ou plusieurs autres arguments. En d'autres termes, l'acceptabilité d'un argument A représente sa validité. Elle est justifiée grâce aux relations existantes entre A et les arguments en sa faveur. L'ensemble des arguments acceptables, i.e. dont la validité n'est pas remise en cause, est dit admissible pour un agent à la condition que cet ensemble soit libre de conflit. Les arguments d'un tel ensemble sont alors soutenus par l'agent rationnel.

Définition 3.2.II [Libre de conflit - Acceptabilité - Admissibilité]

> Soient $AF = \langle AR, attacks \rangle$ un cadre d'argumentation, $S \subseteq AR$ un ensemble d'arguments et $A \in AR$ un argument.
> 1. S est **libre de conflits** ssi $\forall A, B \in S, \neg attacks(A, B)$;
> 2. A est **acceptable** vis à vis de S ssi $\forall B \in AR, attacks(B, A) \Rightarrow attacks(S, B)$;
> 3. S libre de conflits est **admissible** ssi $\forall B \in S$, B est acceptable vis-à-vis de S.

Un ensemble d'arguments admissible et maximal est appelé extension préférée.

Définition 3.2.III [Extension préférée]

> Soient $AF = \langle AR, attacks \rangle$ un cadre d'argumentation. Une *extension préférée* du cadre d'argumentation AF est un ensemble admissible d'arguments $S \subset AR$ qui est maximal (au sens de l'inclusion).

Tout cadre d'argumentation possède au moins une extension préférée pouvant être éventuellement vide. La rationalité acceptant les arguments d'au moins une extension préférée est dite crédule [30]. Tout agent manipulant une telle rationalité est qualifié de crédule.

Définition 3.2.IV [Fonction caractéristique]

Soit $AF = \langle AR, attacks \rangle$ un cadre d'argumentation. La *fonction caractéristique* associée au cadre d'argumentation *AF*, notée F_{AF}, est définie comme suit :

$$F_{AF} : 2^{AR} \rightarrow 2^{AR}$$

$$F_{AF}(S) = \{A \in AR \mid A \text{ est acceptable vis-à-vis de } S\}.$$

F_{AF} est monotone au sens de l'inclusion ensembliste, i.e. un argument acceptable vis-à-vis de *S* l'est également pour tout sur-ensemble de S.

L'extension raisonnable est définie comme étant le point fixe de la fonction caractéristique.

Définition 3.2.V [Extension raisonnable]

Soient $AS = \langle AR, attacks \rangle$ un cadre d'argumentation. L'*extension stable* du système d'argumentation *AS*, notée GE_{AS}, est le point fixe de F_{AS}.

Un agent est sceptique [30] lorsqu'il base son raisonnement sur une rationalité raisonnable. Cette rationalité considère les arguments comme acceptables uniquement s'ils font parti de l'extension raisonnable.

L'ensemble de ces termes est illustré dans les exemples suivants :

Exemple 3.2.I [Cadre d'argumentation]

L'ornithorynque est un petit mammifère semi-aquatique, qu'on ne trouve qu'à l'est de l'Australie. C'est le seul mammifère qui pond des œufs au lieu de donner naissance à des petits vivants. Cependant, il ne semble pas évident que l'ornithorynque soit un mammifère. Une personne non familière avec la faune Australienne pourrait penser qu'il s'agit d'un oiseau. En effet, les arguments suivants confortent cette idée :

- C_1 : L'ornithorynque a un bec de canard.

- C_2 : L'ornithorynque pond des oeufs.

Néanmoins, plusieurs arguments permettent de conclure que

l'ornithorynque est un mammifère.

- P_1 : L'ornithorynque n'a pas d'ailes.

- P_2 : L'ornithorynque allaite ses petits.

Ces quatre arguments constituent un cadre d'argumentation $AF = \langle AR, attacks \rangle$ où $AR = \{P_1, P_2, C_1, C_2\}$ et $attacks = \{(P_1, C_1), (C_2, P_1), (P_2, C_2), (P_2, C_1)\}$ dont l'extension préférée est l'ensemble $P = \{P_1, P_2\}$. Cet ensemble représente également

l'extension raisonnable. En effet, *P* est à la fois admissible et le point fixe de la fonction $F_{AF.}$

$$F_{AF}(\emptyset)=\{P_2\},\ \ F^2_{AF}(\emptyset)=\{P_1, P_2\}\ \ et\ \ F^3_{AF}(\emptyset)=F^2_{AF}(\emptyset)$$

La théorie proposée par Dung permet de généraliser les approches en raisonnement non-monotone et en programmation logique. En effet, Dung [30] établit les relations entre les extensions introduites précédemment et celles d'autres formes de raisonnements non-monotone. Néanmoins, Dung considère que l'attaque d'un argument par un autre argument est systématiquement victorieuse. Toutefois, la validité d'un argument dépend de plusieurs facteurs tels que le contexte ou les connaissances dont dispose l'agent. Dans la section suivante, nous présentons un cadre d'argumentation dont l'acceptabilité des arguments est basée sur les préférences entre les arguments.

3.2.2 Cadres d'argumentation à base de préférences

Les cadres d'argumentation à base de préférences, introduits par Amgoud [6][7], reconsidèrent l'acceptabilité des arguments proposée par Dung. Désormais, l'acceptabilité d'un argument ne dépend plus uniquement des arguments (resp. des contre-arguments) qui le défendent (resp. qui l'attaquent) mais également de la force de ces arguments. Pour cela, le cadre d'argumentation (cf. Définition 3.2.VI) est doté d'une relation de préférence qui lie les arguments.

Définition 3.2.VI [Cadre d'argumentation à base de préférences]

Un *cadre d'argumentation à base de préférences* est un triplet $PAF = \langle AR, attacks, pref \rangle$ tel que :
- $\langle AR, attacks \rangle$ est un cadre d'argumentation.
- *pref* est un ordre (partiel ou total) sur $AR \times AR$, appelé relation de préférence.
Pour deux arguments *A* et *B*, la notation *pref*(*A*, *B*) signifie que *A* est préféré à *B*.

La relation de préférence permet de définir deux notions d'acceptabilité. La première, l'acceptabilité individuelle (cf. Définition 3.2.VII), dénote le fait qu'un argument peut se défendre de lui même. C'est-à-dire qu'il résiste aux attaques de ses contre-arguments car il est préféré à eux. La seconde, l'acceptabilité conjointe (cf. Définition 3.2.VIII), dénote le fait qu'un argument a besoin de l'appui d'autres arguments pour être considéré comme acceptable. Par conséquent, l'attaque d'un argument *A* par un contre-argument *B* peut échouer si *A* est préféré à *B*. Cette relation, appelée défaite est définie comme suit :

Définition 3.2.VII [Défaite]

Soient $PAF = \langle AR, attacks, pref \rangle$ un système d'argumentation à base de préférences et $A,B \in AR$ deux arguments. A défait B, noté $defeats(A,B)$ ssi $attacks(A,B) \wedge \neg pref(A,B)$.

Dans la suite, nous dirons plutôt que « A défait B » ou encore que « B est défait par A » si $defeats(A, B)$. Par abus de langage, on dira qu'un ensemble d'arguments S défait B (ou B est défait par S) si B est défait par au moins un argument de S, i.e. $defeats(S,B) \Leftrightarrow \exists A \in S, defeats(A,B)$.

Du fait que l'attaque n'est plus garantie et que son succès dépend désormais de la relation de préférence, la Définition 3.2.II a été redéfinie comme suit :

Définition 3.2.VIII [Libre de conflit – Acceptabilité - Admissibilité]

Soient $PAF=\langle AR, attacks, pref \rangle$ un système d'argumentation à base de préférence, $S \subseteq AR$ un ensemble d'arguments et $A \in AR$ un argument.

1. S est **libre de conflits** ssi $\forall A, B \in S, \neg defeats(A,B)$;
2. A est **acceptable vis-à-vis** de S ssi $\forall B \in AR, defeats(B,A) \Rightarrow defeats(S,B)$;
3. S libre de conflits est **admissible** ssi $\forall B \in S$, B est acceptable vis-à-vis de S.

Dans ce type de cadre, une extension préférée est toujours considérée comme l'ensemble maximal (au sens de l'inclusion) des arguments admissibles. Néanmoins, la transitivité et l'asymétrie de la relation de préférence assurent le fait que, l'extension préférée d'un cadre d'argumentation à base de préférences est non-vide.

L'Exemple 3.2.II reprend le cadre défini dans l'Exemple 3.2.I en précisant la relation de préférence propre au cadre d'argumentation à base de préférences.

Exemple 3.2.II [Cadre d'argumentation à base de préférences]

Nous reprenons ici, le cadre d'argumentation introduit dans l'Exemple 3.2.I. Soit $PAF=\langle AR, attacks, pref \rangle$ un cadre d'argumentation à base de préférences tel que :

- $AR=\{P_1, P_2, C_1, C_2\}$,
- $attacks=\{(P_1,C_1),(C_2,P_1),(P_2,C_2),(P_2,C_1)\}$,
- $pref=\{(C_2,P_2)\}$.

L'extension préférée est désormais l'ensemble $\{C_1, C_2\}$ qui est également l'extension raisonnable. En effet, C_2 est préféré à P_2 et défait avec succès P_2. Par conséquent, C_1 est acceptable.

3.2.3 Cadres d'argumentation à base de valeurs

Les cadres d'argumentation à base de valeurs [13][14] sont également une extension des travaux de Dung [30]. Ils gardent comme principe de lier les arguments entre eux par une relation d'attaque mais, autorisent les agents à se rassembler en groupes distincts. Au sein de ces groupes, appelés audience, les agents partagent des préférences communes. Ces préférences leur permettent de départager les arguments et d'affiner la relation d'attaque introduite précédemment. Afin de modéliser les préférences, le cadre d'argumentation est muni de valeurs et d'une relation *promote* permettant d'associer ces valeurs aux arguments.

Définition 3.2.IX [Cadre d'argumentation à base de valeurs]

> Un *cadre d'argumentation à base de valeurs* est un quintuplet $VAF = \langle AR, attacks, V, promote, \Pi \rangle$ tel que :
> - $\langle AR, attacks \rangle$ est un cadre d'argumentation,
> - V un ensemble fini de valeurs $\{v_1, \ldots, v_n\}$,
> - *promote* : $AR \rightarrow V$ est une fonction qui met en relation les arguments et les valeurs,
> - Π est un ensemble d'audiences $\{\pi_1, \ldots, \pi_n\}$.
>
> On dit qu'un argument A prône la valeur v ssi $promote(A) = v$.

Chaque audience se différencie des autres par ses préférences individuelles. Ces préférences sont définies par un ordre sur V. Par conséquent, il existe potentiellement autant d'audiences que d'ordre sur l'ensemble des valeurs V. A chacune de ces audiences peuvent être associées un cadre d'argumentation spécifique.

Définition 3.2.X [Cadre d'argumentation spécifique à une audience]

> Un *cadre d'argumentation spécifique à une audience* est un quintuplet $AVAF = \langle AR, attacks, V, promote, >_{\pi_1} \rangle$ tel que :
> - $\langle AR, attacks, V, promote, \Pi \rangle$ est un cadre d'argumentation à base de valeurs,
> - $\pi_1 \in \Pi$ une audience,
> - $>_{\pi_1} \subseteq V \times V$ est la relation de préférences de l'audience π_1.
>
> Soient $v_1, v_2 \in V$, $v_1 >_{\pi_1} v_2$ se lit v_1 est préféré à v_2 selon l'audience π_1.

A partir de cette définition, on est amené à redéfinir la notion de défaite (cf.Définition 3.2.XI) pour inclure l'audience concernée.

Définition 3.2.XI [Défaire pour une audience]

Soient $AVAF = \langle AR, attacks, V, promote, >_{\pi_1} \rangle$ un cadre d'argumentation spécifique à une audience π_1, $A, B \in AR$ deux arguments. *A défait B selon une audience* π_1, noté $defeats_{\pi_1}(A, B)$, ssi :

$$attacks(A, B) \wedge \neg(promote(B) >_{\pi_1} promote(A)) .$$

Dans le cas où deux arguments A et B promeuvent la même valeur (ou aucune valeur), l'attaque est considérée comme étant menée avec succès, i.e. :

$$defeats_{\pi_1}(A, B) \Leftrightarrow attacks(A, B) \wedge (promote(A) = promote(B)) .$$

Par abus de langage, on dira qu'un ensemble d'arguments *S* défait *B* selon l'audience π_1 ssi $defeats_{\pi_1}(S, B) \Leftrightarrow \exists A \in S, defeats_{\pi_1}(A, B)$.

En général, une valeur de l'ensemble *V* est attribuée à plusieurs arguments. Toutefois, si chaque argument de *AR* promeut une valeur de *V* différente, le cadre d'argumentation *AVAF* est alors considéré comme un cadre d'argumentation à base de préférences (cf. Définition 3.2.VI). A l'inverse, si tous les arguments de *AR* promeuvent la même valeur de *V*, le cadre d'argumentation *AVAF* est considéré comme un cadre d'argumentation abstrait (cf. Définition 3.2.I). Par conséquent, les notions d'acceptabilité et d'admissibilité (cf. Définition 3.2.XII) peuvent être adaptées comme suit.

Définition 3.2.XII [Libre de conflits – Acceptabilité – Admissibilité]

Soit $AVAF = \langle AR, attacks, V, promote, >_{\pi_1} \rangle$ un cadre d'argumentation spécifique à une audience π_1. Soient *A*, *B* $\in$ *AR* deux arguments et *S* $\subseteq$ *AR* un ensemble d'arguments.

1. *S* est **libre de conflits** selon l'audience π_1 ssi $\forall A, B \in S \ \neg defeats_{\pi_1}(A, B)$,
2. *A* est **acceptable selon l'audience** π_1 vis-à-vis de *S* ssi :
$$\forall B \in AR, defeats_{\pi_1}(B, A) \Rightarrow defeats_{\pi_1}(S, B)$$
3. *S* libre de conflits est **admissible selon l'audience** π_1 ssi $\forall A \in S$, *A* est acceptable selon l'audience π_1 vis-à-vis de *S*.

L'extension préférée selon l'audience π_1 du cadre d'argumentation *AVAF* est un ensemble admissible selon l'audience π_1 qui est maximal au sens de l'inclusion (cf. Définition 3.2.III).

Dans les cadres d'argumentation à base de préférences, la notion d'acceptabilité (cf. section 3.2.2) a été introduite pour permettre à un groupe d'agents (i.e. appartenant à une unique audience) de départager un ensemble d'arguments à l'aide de préférences pré-établies. L'introduction des audiences multiples permet de distinguer trois types d'acceptabilités : i)

l'acceptabilité objective est l'accord unanime accordé par l'ensemble des audiences de Π sur le fait qu'un argument soit valide, ii) l'acceptabilité subjective admettant qu'un argument est valide pour un certain nombre d'audiences et iii) l'acceptabilité indéfendable qui admet qu'un argument peut être soutenu par aucune des audiences de Π.

Définition 3.2.XIII [Acceptabilité objective – subjective – indéfendable]

Soit $VAF = \langle AR, attacks, V, promote, \Pi \rangle$ un cadre d'argumentation à base de valeurs. Soient $A \in AR$ un argument et $S \subseteq AR$ un ensemble d'arguments :

1. A est *acceptable objectivement* vis-à-vis de S ssi $\forall\ \pi_i \in \Pi$, A est acceptable selon l'audience π_i *vis-à-vis* de S.
2. A est *acceptable subjectivement* vis-à-vis de S ssi $\exists\ \pi_i \in \Pi$, A est acceptable selon l'audience π_i *vis-à-vis* de S.
3. A est *indéfendable* vis-à-vis de S ssi A est ni acceptable objectivement ni acceptable subjectivement vis-à-vis de S.

L'exemple suivant reprend le cadre d'argumentation présenté dans l'Exemple 3.2.I en spécifiant la relation *promote,* l'ensemble des valeurs et la relation de préférences des audiences π_1 et π_2 .

Exemple 3.2.III [Cadre d'argumentation à base de valeurs]

Soient $AF_{\pi_1} = \langle AR, attacks, V, promote, \succ_{\pi_1} \rangle$ et $AF_{\pi_2} = \langle AR, attacks, V, promote, \succ_{\pi_2} \rangle$ les cadres d'argumentation spécifiques des audiences π_1 et π_2 tels que :

- $AR = \{P_1, P_2, C_1, C_2\}$,
- $attacks = \{(P_1, C_1), (C_2, P_1), (P_2, C_2), (P_2, C_1)\}$,
- $V = \{mammifère, oiseau\}$,
- $promote = \{(C_1, oiseau), (C_2, oiseau), (P_1, mammifère), (P_2, mammifère)\}$,
- $mammifère \succ_{\pi_1} oiseau$ et $oiseau \succ_{\pi_1} mammifère$

Dans ce contexte, les ensembles $\{P_1, P_2\}$ et $\{C_1, C_2\}$ sont respectivement les extensions préférées des audiences π_1 et π_2 .

3.2.4 Bilan

Dans un souci de généricité, les cadres d'argumentation présentés précédemment considèrent uniquement les relations entre les arguments. Principalement, les arguments sont munis d'une relation de contradiction, appelée attaque. Associée à des relations de priorité

prenant en compte les préférences individuelles d'un ou plusieurs agents, la relation d'attaque permet de déterminer l'acceptabilité d'un argument. La notion d'acceptabilité permet, pour un agent rationnel, de déterminer si un argument doit être accepté ou rejeté.

Dans cette section, les arguments sont des entités abstraites. Certains travaux s'intéressent, cependant à associer une structure aux arguments. L'introduction d'une telle structure permet alors d'enrichir les relations précédemment introduites. Un cadre d'argumentation dans lequel la structure des arguments est définie à l'aide d'un langage logique est appelé logique argumentative.

3.3 Logiques argumentatives

Au niveau des logiques argumentatives, les arguments sont formalisés par le biais d'un langage logique $\mathcal{L}$. Désormais, un argument est un ensemble de règles logiques qui, mises ensemble permettent de conclure à la validité (ou l'invalidité) d'une proposition. Ce formalisme permet d'introduire différentes relations de contradiction pouvant être basées sur la structure des arguments ou sur les préférences qui leur sont attribuées.

Dans un premier temps, nous introduisons la structure des arguments. Puis, nous présentons les différentes relations de contradiction en commençant par les travaux de Schroeder (cf. section 3.3.2). Nous nous intéressons ensuite, aux logiques argumentatives dans lesquelles une notion de force a été définie (cf. section 3.3.3 à 3.3.6).

3.3.1 Arguments

Suivant les travaux (cf. section 3.3.2 à 3.3.6), le langage $\mathcal{L}$ utilisé pour définir la structure interne des arguments est différent. Néanmoins, les arguments peuvent tous être représentés de manière général comme un couple (P, c) exprimant le fait que c peut-être déduit logiquement de P. P est appelé prémisse de l'argument et c est sa conclusion. La prémisse de l'argument est composée de règles bien formées construites sur $\mathcal{L}$. Ces règles sont regroupées dans une base de connaissances ou un programme logique Σ. La conclusion de l'argument, quant à elle, est une formule du langage $\mathcal{L}$. Dans la suite, on dénote par $\mathcal{A}(\Sigma)$ l'ensemble des arguments construits sur Σ.

Définition 3.3.I [Argument]

Un *argument* est un couple $A = (P, c)$ où c est une formule logique et P un sous-ensemble de Σ tel que :
1. $P \subseteq \Sigma$ est consistant,
2. $P \vdash c$,
3. P est minimal, i.e. il n'existe pas de sous-ensemble de P vérifiant 1. et 2.

P est appelé prémisse de A, notée $P=premise(A)$ et c est la conclusion de A, notée $c=conclusion(A)$.

3.3.2 Relations de contradiction

Schroeder s'intéresse à identifier différentes notions de contradiction. Dans [109], Il propose l'utilisation de programmes de logique étendue pour formaliser la structure des arguments. L'avantage de tels programmes est de pouvoir modéliser les deux notions fondamentales d'attaques [91] : la sape et la réfutation. Ces notions fondamentales sont considérées pour définir de nouvelles relations d'attaque.

Définition 3.3.II [Relations de contradiction]

Soient A et B deux arguments. Une relation de contradiction est une relation binaire définie sur le domaine $A(\Sigma) \times A(\Sigma)$ telle que :

1. A sape B (*undercuts*) ssi il existe un littéral objectif[5] L tel que $L = conclusion(A)$ et $notL$[6] est inclus dans $premise(B)$;
2. A réfute B (*rebuts*) ssi il existe un littéral objectif L tel que L est une conclusion de A et $\neg L$ est une conclusion de B;
3. A attaque B (*attacks*) ssi A sape B ou A réfute B;
4. A défait B (*defeats*) ssi A sape B, ou A réfute B sans que B sape A;
5. A attaque vigoureusement B (*strongly attacks*) ssi A attaque B et B ne sape pas A;
6. A sape vigoureusement B (*strongly undercuts*) ssi A sape B et B ne sape pas A.

Nous pouvons constater que l'ajout d'une structure aux arguments permet d'introduire des relations de contradiction plus fines. Contrairement aux cadres d'argumentation, les arguments sont liés entre eux par les éléments constituant leur prémisse et leur conclusion. Les travaux suivants s'intéressent à munir les arguments non seulement d'une structure mais également d'une force.

[5] Un littéral objectif est un atome A ou sa négation explicite $\neg A$.

[6] *notL* signifie : « Il n'y a aucune raison de croire en L ».

3.3.3 Relations de spécificité

La logique argumentative définie par Simari et Loui [114] détermine l'acceptabilité d'un argument en utilisant la notion de spécificité proposée Poole [88].

Définition 3.3.III [Spécificité]

Soient $\mathcal{L}_f$ l'ensemble des formules de $\mathcal{L}$ et Σ_n l'ensemble des formules non fondées de Σ. Soit $A=(P,c)$ et $B=(P',c')$ deux arguments. On dit que A est *strictement plus spécifique* que B, noté $A \succ_{spec} B$ ssi :

1. $\forall l \in \mathcal{L}_f$ tel que $\Sigma_n \cup \{l\} \cup P \vdash c$ et $\Sigma_n \cup \{l\} \cup P' \nvdash c$ avec $\Sigma_n \cup \{l\} \cup P' \vdash c'$ ou,
2. $\exists l \in \mathcal{L}_f$ tel que $\Sigma_n \cup \{l\} \cup P' \vdash c'$ et $\Sigma_n \cup \{l\} \cup P \nvdash c$ en tenant compte que c' ne découle pas indépendamment de P', i.e. $\Sigma_n \cup \{l\} \nvdash c'$.

En clair, un argument A est dit plus spécifique qu'un argument B à partir du moment où A peut être appliqué moins souvent que B. Lors d'une contradiction entre deux arguments, on considère comme acceptable celui qui est le plus spécifique. Désormais, on peut redéfinir la relation de défaite (cf. Définition 3.2.VII).

Définition 3.3.IV [Défaite]

Soit A et B deux arguments. On dit que A *défait* B ssi :

1. A est un contre-argument de B, i.e. A attaque B (cf. Définition 3.3.II),
2. $A \succ_{spec} B$, i.e. A est plus spécifique que B.

Nous pouvons constater que la défaite n'est pas uniquement établie à l'aide de la structure. Les arguments sont également départagés grâce à la relation de spécificité. La spécificité représente, en quelque sorte, la force d'un argument. Dans la section suivante, nous présentons la logique argumentative basée sur le cadre d'argumentation à base de préférences proposée par Amgoud (cf. 3.2.2). Dans cette dernière, la relation de priorité est assimilée à la notion de préférence.

3.3.4 Relations de priorité statiques

Dans [6][7], Amgoud considère la base de connaissances Σ comme une union d'ensembles disjoints Σ_1, ..., Σ_n ordonnés selon une relation de priorité $\ll$. Ainsi, Σ_1 contient les règles les plus préférées de Σ et Σ_n est l'ensemble des règles les moins préférées. Les règles d'un même ensemble Σ_i sont dites équi-préférées.

Définition 3.3.V [Niveau de préférence]

> Le niveau de préférence d'un ensemble $P \subseteq \Sigma$, noté *level*(*P*), est l'indice maximal de l 'ensemble $\Sigma_i \subseteq \Sigma$ contenant un élément de *P*.

La relation de priorité $\ll$ permet d'attribuer aux arguments un niveau de préférence. Du fait que la base de connaissances Σ soit inconsistante, des arguments contradictoires peuvent être construits en utilisant les règles de Σ. Le niveau de préférence permet alors de départager de tels arguments. On définit ainsi la préférence d'un argument.

Définition 3.3.VI [Préférence d'un argument]

> Soient *A* et *B* deux arguments de $A(\Sigma)$. *A est préféré à B,* noté *pref(A,B)* ssi :
> $$level(premise(A)) > level(premise(B))$$

Les inconsistances de la base de connaissances Σ peuvent être explicites ou déduites. Elles sont dites explicites si deux formules de Σ sont contradictoires, par exemple si p et $\neg$p appartiennent toutes deux à Σ. A l'inverse, elles sont déduites si un argument de $A(\Sigma)$ contredit une formule de Σ. Contrairement à la Définition 3.3.II, la relation de sape introduite par Amgoud ne considère pas la notion de croyance (prédicat *not*), cette dernière est redéfinie comme suit :

Définition 3.3.VII [Sape]

> Soient *A* et *B* deux arguments de $A(\Sigma)$. *A sape B* ssi $\exists c \in premise(B)$ tel que $c \equiv \neg conclusion(A)$. On note *A undercut B.*

En d'autres termes, un argument *A* sape un argument *B* si la conclusion de *A* contredit une formule de la prémisse de *B*.

Ensemble, les relations de sape et de préférence forment une logique argumentative à base de préférences.

Définition 3.3.VIII [Logique argumentative à base de préférences]

Une logique argumentative à base de préférences est un triplet $PAF = \langle A(\Sigma), undercut, pref \rangle$ tel que :
- $A(\Sigma)$ est l'ensemble des arguments contruits à partir de Σ ;
- $undercut \subseteq A(\Sigma) \times A(\Sigma)$ est une relation de contradiction entre arguments (cf. Définition 3.3.VII)
- *pref* est un ordre (partiel ou complet) sur $A(\Sigma) \times A(\Sigma)$. (cf. Définition 3.3.VI)

La logique argumentative présentée ici utilise de façon complémentaire la notion de sape et de préférences afin de déterminer l'acceptabilité d'un argument. Néanmoins, les préférences sont définies à l'aide de Σ et sont par conséquent statiques. Elles ne peuvent ni évoluer ni être remises en cause au cours du raisonnement. La dynamicité des préférences est l'objet des travaux introduits dans la section suivante.

3.3.5 Relations de priorité dynamiques

Dans [55][75][29], Kakas et Moraitis introduisent une relation de priorité dynamique dépendant d'un ensemble d'événements particuliers. Ils considèrent que la relation de priorité $\ll$ définit la politique de décision de l'agent, i.e. les préférences qu'il attribue aux arguments. Cependant, l'agent peut, suivant l'apparition d'événements particuliers, modifier son raisonnement et reconsidérer son jugement. Ces événements forment ce qu'on appelle le contexte spécifique. Dans leur cas, la notion d'argument est définie à l'aide de programme logique sans négation par l'échec [56].

Définition 3.3.IX [Théorie argumentative]

Une *théorie argumentative* d'un agent est un triplet $AT = \langle \Sigma, \ll, priority \rangle$ où :
- Σ est une base de connaissances;
- $\ll$ est une relation de priorité définie sur $\Sigma \times \Sigma$.
- *priority* est une relation de priorité décrivant le contexte spécifique.

La relation d'attaque (cf. Définition 3.2.I) est désormais fondée sur les contradictions possibles entre deux arguments et les relations de priorité.

Définition 3.3.X [Attaque]

Soient $AT = \langle \Sigma, \ll, priority \rangle$ une théorie argumentative et $A=(P,c)$, $B=(P',c')$ deux arguments de $\mathsf{A}(\Sigma)$. *A attaque B* ssi $\exists\ L \in \Sigma$, $P_1 \subseteq P$, $P_2 \subseteq P'$ tels que :

1. $P_1 \vdash L$ et $P_2 \vdash \neg L$;
2. $(\nexists\ P_1' \subset P_1 \text{ tel que } P_1' \vdash L) \wedge (\nexists\ P_2' \subset P_2 \text{ tel que } P_2' \vdash \neg L)$,
3. $\exists r \in P_1, \exists r' \in P_2$ tels que $(r \ll r' \vee priority(r', r)) \Rightarrow \exists s \in P_1, \exists s' \in P_2$ tels que $(s' \ll s \vee priority(s, s'))$.

Ainsi, un argument *A* contredit un argument *B* si les conclusions de *A* et *B* sont contradictoires (condition 1.), s'ils sont minimaux (condition 2.) et que *A* est prioritaire sur *B* (condition 3.). La relation d'attaque introduit dans la Définition 3.3.X est équivalente à la relation de défaite (cf. Définition 3.2.VII).

Exemple 3.3.I [Relation de priorité dynamique]

L'exemple présenté ici s'intéresse de nouveau à l'ornithorynque. Comme précisé dans l'Exemple 3.2.I, cet animal possède les caractéristiques suivantes : il allaite ses petits pond des oeufs et possède un bec de canard. Ces caractéristiques sont décrites respectivement par les règles r_1, r_2 et r_3. Par observation, on a également constaté que l'ornithorynque était un animal semi-aquatique (règle r_4).

- $r_1 : allaite(ornithorynque)$,
- $r_2 : pond(ornithorynque)$,
- $r_3 : bec(ornithorynque)$,
- $r_4 : semi\text{-}aquatique(ornithorynque)$.

De plus, les naturalistes sont unanimes pour dire que la principale caractéristique des mammifères est d'allaiter leur petit. Alors que les oiseaux sont identifiables par leur bec et le fait qu'ils pondent des oeufs. Les règles r_5 et r_6 transcrivent le savoir des naturalistes.

- $r_5 : \forall X, mammifère(X) \leftarrow allaite(X)$,
- $r_6 : \forall X, oiseau(X) \leftarrow pond(X) \wedge bec(X)$.

Étant donné que l'animal étudié pond des oeufs, on considère que la règle r_6 et plus pertinente que la règle r_5, on le note $r_5 \ll r_6$. Cependant, la plupart des oiseaux volent, un animal semi-aquatique a donc plus de chance d'être identifié comme un

mammifère. Cette connaissance forme le contexte spécifique et est décrite de la façon suivante : $priority(r_5, r_6) \leftarrow semi\text{-}aquatique(X)$.

D'après la théorie argumentative formée des règles r_1 à r_6, on peut déduire que l'ornithorynque est un oiseau. En effet, l'argument C=({r_2, r_3, r_6}, oiseau) soutient cette hypothèse. Sachant que l'ornithorynque est un animal semi-aquatique, on peut également conclure que c'est un mammifère. L'argument P=({r_1, r_5, r_4, *priority*(r_5, r_6)}, mammifère) conforte cette idée. Selon le contexte spécifique, l'argument P est préféré a C.

3.3.6 Relation de priorité agent

Morge [77] propose une logique argumentative inspirée du cadre d'argumentation présenté dans la section 3.2.3. Les agents, organisés en audiences, donnent leur avis en fonction de croyances communes et de croyances personnelles. Chaque agent possède une base de connaissances $\Sigma = \Sigma_c \cup \Sigma_p$ composée des croyances communes Σ_c et de ses croyances personnelles Σ_p propres à un agent. Les croyances personnelles sont exposées aux autres agents lors de l'argumentation.

Définition 3.3.XI [Théorie argumentative multi-agents]

Soit $\upsilon_A = \{ag_1, \ldots, ag_n\}$ un ensemble d'agents rationnels. Une **théorie argumentative multi-agents** est un triplet $AT_{\upsilon_A} = \langle \Sigma, V, promote \rangle$ où :

- $\Sigma = \Sigma_c \cup \Sigma_p$ est une base de connaissances telle que :

  $\Sigma_P = \bigcup_{i=1}^{n} \Sigma_{P_i}$ est l'ensemble des croyances des agents de υ_A,
- V est un ensemble de valeurs {v_1, …, v_t},
- $promote : \Sigma \rightarrow V$ est une fonction bijective qui associe une valeur aux arguments.

Soit $A \in \mathsf{A}(\Sigma)$ l'ensemble des arguments construits sur Σ, on dit que l'argument A prône une valeur v ssi $promote(A) = v$.

Comme dans [6][7], la base de connaissances est stratifiée en un ensemble de couches. Chaque couche regroupe les arguments qui prônent une valeur similaire. Par la suite, ces valeurs sont ordonnées selon un ordre propre à chaque agent de υ_A. Les agents raisonnant différemment les uns des autres, on les considère comme une audience. C'est pour cette raison qu'on leur attribue une théorie argumentative spécifique.

Définition 3.3.XII [Théorie argumentative mono-agent]

Soit $ag_i \in \mho_A$ un agent. La théorie argumentative relative à l'agent ag_i est un quadruplet $AT_i = \langle \Sigma_C \cup \Sigma_{P_i}, V, promote, \ll_i \rangle$ où :
- $\langle \Sigma, V, promote \rangle$ est une théorie argumentative multi-agents avec $\Sigma_C \cup \Sigma_{P_i} \subseteq \Sigma$;
- $\ll_i$ est l'ordre sur l'ensemble V relatif à l'agent ag_i.

Les règles r_1 et r_2 sont dites équi-prioritaires si $promote(r_1)=promote(r_2)$.

L'ordre $\ll_i$ est la relation de priorité de l'agent ag_i, il lui permet d'ordonner les valeurs de l'ensemble V. Cette relation retranscrit l'opinion de l'agent sur les formules de Σ. Toutefois, on impose aux agents de $\mho_A$ d'avoir les mêmes opinions sur les formules appartenant à Σ_c. Un niveau de priorité est également attribué aux arguments, il correspond à la valeur la moins prioritaire prônée par une règle de sa prémisse.

Définition 3.3.XIII [Niveau de priorité]

Soit $AT_i = \langle \Sigma_i, V, promote, \ll_i \rangle$ la théorie argumentative de l'agent ag_i. Le *niveau de priorité* de l'argument $A=(P,c) \in \mathcal{A}(\Sigma)$, noté $level_i(A)$ est défini tel que :

$$level_i(A) = v_i \in V \Leftrightarrow \exists\, r \in P \mid \forall\, r' \in P - \{r\},\ v_i = promote(r) \ll_i promote(r')\ .$$

La théorie peut présenter certaines inconsistances menant à des conclusions contradictoires. La relation de défaite présentée ci-après permet de gérer ces inconsistances.

Définition 3.3.XIV [Défaite]

Soit $AT_i = \langle \Sigma, V, promote, \ll_i \rangle$ la théorie argumentative de l'agent ag_i. Soient $A=(P,c)$, $B=(P',c') \in \mathcal{A}(T)$ deux arguments. *A défait B* pour un agent ag_i, noté $defeats_i(A,B)$ ssi :

1. $\exists L \in \mathcal{L}, \exists P_1 \subseteq P, \exists P_2 \subseteq P' \mid P_1 \vdash L \text{ et } P_2 \vdash \neg L$,
2. $level_i(P_2) \ll_i level_i(P_1)$.

Le succès d'une attaque dépend du niveau de priorité de l'argument. Ce niveau étant relatif à une relation de priorité, les arguments sont appréciés différemment par chaque agent. Les notions d'acceptabilité objective et subjective (cf. Définition 3.2.XIII) sont donc utilisables pour départager les arguments au sein d'une ou de plusieurs audiences.

Exemple 3.3.II [Relation de priorité agent]

Nous considérons ici les règles r_1, r_2, r_3, r_5 et r_6 définies dans l'Exemple 3.3.I. Chacune d'elles est utilisée par deux naturalistes qui exposent leur point de vue. Le Tableau 4 contient les opinions de ces deux naturalistes.

Tableau 4 - Relation de priorité des naturalistes

$\ll_{Nat_1}$	V	Σ	$\ll_{Nat_2}$	V	Σ
$v_2 \ll_{Nat_1} v_1$	v_1	r_5, r_6	$v_3 \ll_{Nat_2} v_1$	v_1	r_5, r_6
	v_2	r_1		v_3	r_2, r_3
$v_3 \ll_{Nat_1} v_2$	v_3	r_2, r_3	$v_2 \ll_{Nat_2} v_3$	v_2	r_1

D'après le Tableau 4, le naturaliste 1 (resp. 2) pense que l'ornithorynque est un mammifère (resp. un oiseau). En effet, les arguments $P=(\{r_1, r_5, r_6\}, mammifère)$ et $C=(\{r_2, r_3, r_5, r_6\}, oiseau)$ confirment leur raisonnement car, d'après la Définition 3.3.XIV, $defeats_1(P,C)$ et $defeats_2(C,P)$.

A l'aide de leur logique argumentative, les agents rationnels peuvent construire et évaluer des arguments. Ils collaborent et aboutissent à un consensus en échangeant les arguments ainsi créés et en évaluant ceux d'autres agents rationnels. Néanmoins, ces échanges doivent être consistants, les agents doivent éviter de se contredire. C'est pour cette raison qu'ont été introduits les tableaux d'engagements (*commitment store*). Ces tableaux sont la base des systèmes dialectiques et permettent d'argumenter en prenant en compte les arguments précédemment cités.

3.4 Système dialectique

Un système dialectique [49] est un dispositif formel au travers duquel les agents communiquent pour prendre collectivement une décision. Les agents doivent respecter certaines règles qui visent à rendre le dialogue conforme à une convention, i.e. à spécifier les échanges autorisés et interdits. Ces règles sont appelées règles dialectiques.

Une règle dialectique peut obliger un agent rationnel à tenir des propos cohérents. L'agent doit, par exemple, éviter de se contredire. Les engagements des agents sont pour cela, consignés dans des tableaux appelés tableaux d'engagements [70].

De ce fait, lorsque deux agents interagissent, ils doivent se convaincre mutuellement de la véracité d'une proposition. Au cours du dialogue, les agents échangent des affirmations ou des rétractations (*assert*) sous la forme d'arguments. Ils peuvent également mettre en doute les

propositions d'autres agents (*challenge*) si ces dernières sont incompatibles avec leurs connaissances. Ces dialogues commencent toujours par une demande de résolution de conflits (*question*). Ces différents actes sont consignés dans les tableaux d'engagements respectifs des agents. Le dialogue et l'évolution des tableaux d'engagements sont dictés par des règles dialectiques [72][5]. La Tableau 5 décrit l'évolution des tableaux d'engagements des agents *P* et *C* (respectivement *CS(P)* et *CS(C)*) en fonction de l'acte que *P* adresse à *C*.

Ainsi, si *P* a un argument acceptable (*S, p*) construit sur ses connaissances personnelles et que *C* conclut $\neg p$ à l'aide de l'argument ($S', \neg p$), ces deux agents vont s'échanger des arguments tant qu'ils pourront soutenir *p*, $\neg p$, *S* ou *S'*. Les échanges se terminent lorsque $\Sigma(P) \cup CS(P) \cup CS(C)$ et $\Sigma(C) \cup CS(P) \cup CS(C)$ fournissent un même argument acceptable approuvé à la fois par *P* et *C*. $\Sigma(P)$ et $\Sigma(P)$ dénotent les bases de connaissances respectives de *P* et *C*.

Tableau 5 - Mise à jour des tableaux d'engagements

Actes	**Condition**	**Réponses**	***CS(P)***	***CS(C)***
question(p)		1. *assert(p)* 2. *assert(¬p)* 3. *question(q)*	$CSi(P) = CSi\text{-}1(P)$	$CSi(C) = CSi\text{-}1(C)$
assert(p)	*p* est acceptable selon *P*	1. *accept(p)* 2. *assert(¬p)* 3. *challenge(p)*	$CSi(P) = CSi\text{-}1(P) \cup \{p\}$	$CSi(C) = CSi\text{-}1(C)$
assert(S)	Chaque élément de *S* est acceptable selon *P*	1. *accept(S)* 2. *assert(¬p)* 3. *challenge(p)*, $p \in S$	$CSi(P) = CSi\text{-}1(P) \cup S$	$CSi(C) = CSi\text{-}1(C)$
accept(p)	*p* est acceptable selon *P*		$CSi(P) = CSi\text{-}1(P) \cup \{p\}$	$CSi(C) = CSi\text{-}1(C)$
accept(S)	Chaque élément de *S* est acceptable selon *P*		$CSi(P) = CSi\text{-}1(P) \cup S$	$CSi(C) = CSi\text{-}1(C)$
challenge(p)		*assert(S)* tel que *S* supporte *p*	$CSi(P) = CSi\text{-}1(P)$	$CSi(C) = CSi\text{-}1(C)$

3.5 Synthèse

Les logiques argumentatives permettent un raisonnement collectif entre plusieurs agents rationnels. Ces interactions forment un dialogue dans lequel les agents échangent des arguments construits sur leurs connaissances individuelles (ses croyances) ou partagées. Ce dialogue perdure tant que les agents n'ont pas abouti à un consensus dépendant de l'acceptabilité des arguments ainsi échangés.

L'acceptabilité permet de savoir si un argument peut se défendre ou non des attaques qui lui sont portées. A cet usage, plusieurs relations de contradiction ont été introduites. Celles-ci ne sont pas suffisantes pour décrire les préférences des agents. C'est pour cette raison que les logiques argumentatives englobent également une relation de priorité décrivant les préférences d'un agent ou d'une audience. Un argument est alors acceptable s'il vérifie les relations de contradiction et de priorité.

Toutefois, les préférences comme les bases de connaissances sont statiques. De ce fait, les préférences ne peuvent ni évoluer, ni être remises en cause par l'ajout ou le retrait de certaines formules des bases de connaissances. Une exception est la relation de priorité dynamique introduite par Kakas (cf. section 3.3.5) qui attribue une préférence selon le contexte de l'agent. Cependant, l'ordre défini par cette relation n'est pas complet, une modification du contexte peut avoir aucune influence sur les préférences de l'agent.

Deux questions se posent alors : i) Comment positionner les connaissances extérieures (relatives au contexte) par rapport aux connaissances individuelles de l'agent ? et ii) Comment prendre en considération la dynamique de la base de connaissances ?
La réponse à la question i) est en partie apportée par les cadres d'argumentations à base de valeurs. En effet, considérer un ensemble de valeurs permet de réordonner les formules de la base de connaissances aisément. Il est cependant nécessaire d'introduire un ordre dynamique sur ces valeurs. La question ii) est pour le moment ouverte.

Conclusion de l'état de l'art

L'élaboration d'un diagnostic pluridisciplinaires nécessitent la participation de multiple experts. Ces experts se doivent de s'échanger mutuellement des informations et des connaissances de leur discipline respective dans le but d'élaborer un diagnostic différentiel ou un modèle pertinent du système observé. Cependant, le manque de connaissances sur les relations inter-disciplinaires

C'est pour cette raison que nous nous sommes intéressés aux techniques couramment utilisées pour l'élaboration d'un diagnostic réparti (cf. Chapitre 2). Un diagnostic réparti est le résultat de la fusion d'un ensemble de diagnostics élaborés au sein d'un composant du système et/ou par plusieurs agents. Ces diagnostics s'appuient soit sur des méthodes d'apprentissage (cf. section 1.2) soit sur un modèle décrivant le fonctionnement partiel ou global du système (cf. section 1.3). L'argumentation (cf. Chapitre 3) est un modèle de raisonnement qui selon différentes rationalités ou acceptabilités permet de gérer et manipuler des propositions ou arguments. L'argumentation, complétée par un modèle dialectique ou dialogisme permet d'échanger des informations entre agent et mettre en place des dialogues de négociation. Le résultat de ces échanges est un argumentaire validant une certaine conclusion. Dans le cas ou cette conclusion correspond au défaut que l'on cherche à diagnostiquer, cet argumentaire peut-être assimilé à un diagnostic réparti. L'argumentation possède l'avantage de travailler à partir des bases de connaissances des agents permettant ainsi d'agréger leur opinion et d'expliquer de manière compréhensible le raisonnement effectué à l'utilisateur.

Nous focalisons notre attention sur l'argumentation. En effet, l'argumentation permet de satisfaire les contraintes inhérentes au sujet de la thèse : i) la présence de plusieurs experts et ii) le manque de connaissances sur le fonctionnement du système. Le premier point nous oblige à avoir un mécanisme de collaboration entre les experts. Et, le second point ne nous permet pas de nous intéresser aux approches de diagnostic à l'aide d'un ou plusieurs modèles car, il nous est impossible de modéliser le fonctionnement du système. L'argumentation étant basée sur la création et l'échange d'arguments, elle incorpore un mécanisme de collaboration et un mécanismes de raisonnement. Son mécanisme de raisonnement s'appuyant sur une base de

connaissances pouvant être apprise ou fournie par les experts, elle satisfait les contraintes i) et ii).

Cependant, l'argumentation et plus précisément les logiques argumentatives nécessitent une relation de défaite (cf. section 3.3) dans le but d'évaluer la pertinence des arguments proposés et décider de leur acceptabilité (cf. section 3.3.2). Cette relation de défaite s'appuie sur la notion d'attaque et une relation d'ordre. La notion d'attaque considère la structure des arguments pour déterminer si deux arguments sont en accord ou se contredisent. Dans le cas ou les arguments se contredisent, la relation d'ordre les départage. Dans le cadre de notre travail, nous souhaitons évaluer un argument selon un contexte précis. Ce contexte est décrit par un ensemble de faits (i.e. les règles sans prémisse) extérieurs à la base de connaissances. Par conséquent, la relation d'ordre doit être à la fois dynamique, totale et non arbitraire sur les règles de la base de connaissances. Hormis dans [55][56], ou des préférences préfixées sont attribuées aux règles, les logiques argumentatives n'offrent pas cette possibilité. **Ainsi, nous souhaitons disposer d'une logique argumentative dont la relation d'ordre est dynamique, totale et non arbitraire.**

Par ailleurs, les agents mettent à jour leur base de connaissances en fonction des arguments échangés durant l'argumentation (cf. section 3.4). Ils sont ainsi capable de remettre en question les arguments d'autres agents et/ou de les utiliser pour poursuivre leur raisonnement. Toutefois, nous ne souhaitons pas conserver d'arguments qui sont en contradiction avec la base de connaissances de l'agent. En effet, ces connaissances sont issues de l'expérience personnelles d'un expert, il nous semble important de conserver cette expertise. Cependant, il est important que les agents puissent intégrer l'avis d'autres agents dans leur raisonnement. **Ainsi, nous souhaitons disposer d'une mémoire partagée**. Malgré cette mémoire partagée, il est difficilement envisageable de faire argumenter des agents possédant une expertise sur des disciplines distinctes. En effet, les agents en question ne pourront pas proposer d'arguments en faveur ou défaveur des arguments précédemment énoncés. **Ainsi, nous souhaitons disposer d'un média de communication entre les domaines**. À notre connaissance, aucun cadre d'argumentation ne répond à ces deux exigences.

En résumé, le modèle présenté dans la partie suivante est un **outil de diagnostic basé sur une logique argumentative** dont la relation d'ordre reflète l'expérience de l'expert. Cette relation d'ordre devant prendre en considération l'évolution du système, se doit d'être dynamique, totale et non-arbitraire. De plus, les agents doivent partager leur avis sans pour autant intégrer le(s) argument(s) qui en sont à l'origine. C'est pour cette raison que nous fournissons aux agents **un média de communication inter et intra-domaines**.

Deuxième partie

Modèle ANDi

Introduction du modèle

Afin d'élaborer un diagnostic pluridisciplinaires à partir d'un ensemble de faits, nous proposons le modèle ANDi. Ce modèle est un système de diagnostic distribué hiérarchique basé sur une logique argumentative et un système multi-agents. Chaque agent appartient à une unique discipline (appelée domaine) et possèdent sa propre base de connaissances. Les bases de connaissances ainsi attribuées aux agents sont construites à l'aide de méthodes de classification (cf. Chapitre 1) et d'un ensemble de cas. On appelle cas un ensemble de données relative à un problème du système observé accompagné de sa solution. Le modèle ANDi comporte trois niveaux distincts :

1. un niveau inférieur : les arguments construits par les agents à partir de leur base de connaissance et d'un logique argumentative probabiliste (cf. Figure 1) ;
2. un niveau intermédiaire : les superviseurs qui gèrent les échanges entre les agents d'un même domaine dans le but de construire les diagnostic locaux ;
3. un niveau supérieur : le réseau Bayésien qui établit un diagnostic global du système en utilisant les conclusions des diagnostics locaux.

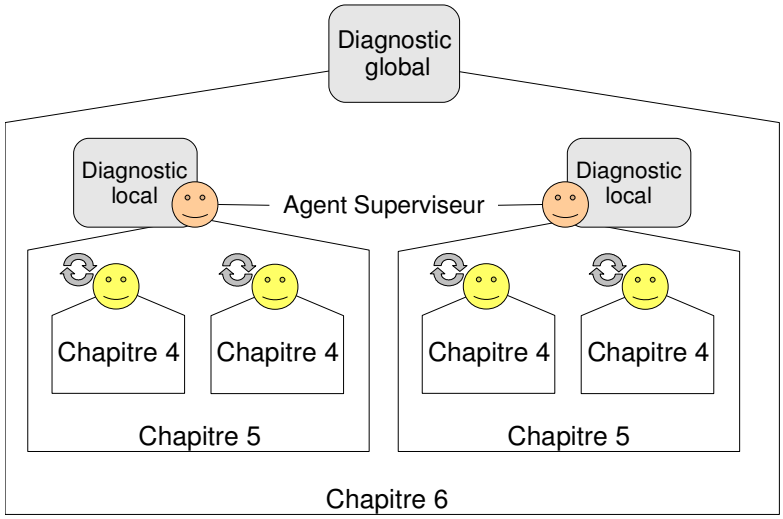


Figure 1 - Organisation de la Partie II

Le coeur du raisonnement est une logique argumentative probabiliste (cf. Figure 1). Cette logique argumentative, présentée dans le Chapitre 4, permet aux agents de construire des arguments à partir des connaissances des experts humains qu'ils représentent dans le système. Dans le cas où ces arguments sont contradictoires, elle leur permet également d'évaluer la pertinence des ces derniers. Pour cela, nous attribuons une force aux arguments dépendant à la fois i) d'une la base de cas et ii) de la base de connaissances.

En échangeant de tels arguments, les agents constituent un argumentaire ayant pour but d'identifier précisément le problème et ses origines. Afin d'éviter les erreurs de diagnostic, nous souhaitons avoir un diagnostic complet, i.e. un diagnostic pour lequel toutes les propositions des agents ont été (in)validées. Par ailleurs, le fait que le problème soit pluridisciplinaire, implique que les agents (au même titre que les experts humains) appartiennent à des domaines d'expertise différents. Il est par conséquent difficile, pour deux agents de domaines distincts, d'évaluer la force des arguments de leur interlocuteur. C'est pour cette raison que nous déléguons à des agents, nommé Agents Superviseur, la direction des débats (cf. Figure 1, *Agent Superviseur* et *Diagnostic local*). Un diagnostic, dit local, est alors créé sur chaque domaine d'expertise. Nous détaillons, dans le Chapitre 5 comment un diagnostic local est élaboré.

Par la suite, les conclusions de ces diagnostics locaux sont utilisées pour proposer un diagnostic plus général. Ce diagnostic, appelé diagnostic global (cf. Figure 1, *Diagnostic global*), considère les relations entre les domaines pour affiner les diagnostics locaux précédemment réalisés. Étant donné que ces relations sont méconnues, voire inconnues des experts humains, nous avons décidé de les découvrir à l'aide de méthodes d'apprentissage. Le Chapitre 6 détaille la construction du modèle statistique et l'utilisation de ce modèle.

Chapitre 4

Cadre d'argumentation probabiliste

Sommaire

4.1 Introduction

Dans ce chapitre, nous proposons une logique argumentative constituant le raisonnement des agents. Cette logique argumentative se base sur des connaissances déduites d'un ensemble de cas, pour gérer les interactions entre des arguments contradictoires. Chaque cas est un ensemble d' « attributs - valeurs » (les attributs sont identiques pour chaque cas étudié) à partir desquels des modèles peuvent être élaborés.

Comme on a pu le voir dans le Chapitre 3, une logique argumentative est caractérisée par l'attribution d'une structure aux arguments. Les arguments sont alors reliés entre eux par des relations de contradiction portant sur cette structure ainsi qu'une relation de priorité permettant à l'agent d'affirmer son inclinaison pour tel ou tel argument.

Dans ce chapitre, on introduit une logique argumentative en définissant chacun des points cités précédemment. Nous commencerons par définir la structure des arguments (cf. section

3.3.1). Ensuite, nous introduirons la notion de force qui nous permet d'ordonner de tels arguments (cf. section 4.3). Après avoir introduit les relations de contradiction (cf. section 4.4), nous proposerons une modélisation de cette logique sous la forme d'un graphe appelé graphe argumentatif (cf. section 4.5).

4.2 Arguments

Les connaissances sont un ensemble de règles décrivant partiellement ou totalement le domaine sur lequel on travaille. Ces connaissances peuvent être issues d'experts humains ou de méthodes de classification. Généralement, les experts humains établissent de telles règles à partir de leur expérience, i.e. des cas qu'ils ont observés. Ces cas sont un ensemble de couples $\langle attribut, valeur \rangle \in \langle Att(L), Att(\Sigma), U_a \rangle$. Afin d'exprimer ces éléments, nous définissons le langage L relatif au domaine considéré.

Définition 4.2.I [Langage]

Soit Σ un alphabet, le langage L_D est un ensemble de termes de Σ^* employés pour décrire un domaine *D*.
On dénote par Att(L_D), l'ensemble des termes de L faisant référence à un attribut du domaine. Soit $a \in Att(L_D)$, on désigne l'ensemble de définition de *a* (i.e. l'ensemble des valeurs possible de *a*) par U_a.
On désigne par $U = \{U_a \mid a \in Att(L_D)\}$ l'ensemble représentant l'union des U_a.

Exemple 4.2.I [Langage]

Soit un langage L tel que :

L= {défaut, matière, cambrage, mal-garnie, acier, aluminium, carbone}.

L'ensemble des attributs du langage L est Att(L) ={défaut, matière} avec :

$$U_{défaut} = \{\text{cambrage, mal-garnie}\} \text{ et}$$
$$U_{matière} = \{\text{acier, aluminium, carbone}\}.$$

Nous définissons désormais une règle argumentative (cf. Définition 4.2.II) comme étant une règles d'associations (cf. Chapitre 1).

Définition 4.2.II [Règle argumentative]

Soient un langage L_D et $Att(L_D)$ un ensemble d'attributs du langage L_D. Une règle argumentative est définie comme une conjonction de couples (attribut, valeur) conduisant à la production d'un nouveau couple.

$$r: \ (a_{n+1}, v_{n+1}) \ \leftarrow \ \bigwedge_{i=1}^{n} (a_i, v_i) \ \text{ ou } \ \forall i, j \in [1,\ldots,n+1] a_i \in Att(L), v_i \in U_{a_i} \text{ et } \forall a_j \neq a_i$$

Le couple (a_{n+1}, v_{n+1}) constitue la *tête* de la règle argumentative *r*, notée *head(r)*.
L'expression composée des autres attributs et de leur valeur est appelée *corps* de la règle argumentative *r*, on le note *body(r)* :

$$body(r) = \bigwedge_{i=1}^{n} (a_i, v_i)$$

Dans la suite, on dira que le couple $(a, v) \in Att(L_D) \times U_a$ appartient à *body*(*r*) ssi :

$$\exists j \in [1,\ldots,n], \ body(r) = \bigwedge_{i=1}^{k} (a_i, v_i) \wedge (a_j, v_j) \bigwedge_{k=(j+1)}^{n} (a_k, v_k) \ \text{ tq } (a, v) = (a_j, v_j)$$

Les valeurs a_i et v_i du couple $c=(a_i, v_i)$ sont respectivement accessibles par les méthodes *c.attribute* = a_i et *c.value* = v_i.

Exemple 4.2.I [Règle argumentative]

Dans une forge, des pièces métalliques sont usinées par trois équipes de forgerons (novice, intermédiaire et experte). Ces pièces sont caractérisées par leurs dimensions (épaisseur et largeur) et leur matière. Ces caractéristiques influent sur le coût et la complexité de l'usinage. Ainsi, les pièces coûteuses ou complexes sont réalisées par l'équipe la plus expérimentée. Elles sont principalement caractérisées par des côtes importantes (une largeur ou une épaisseur importante) ou composées de métaux tels l'aluminium ou le carbone. Malgré l'expérience de l'équipe, quelques défauts subsistent. La pièce peut-être mal garnie (les bords sont concaves) ou cambrée. Les règles argumentatives, décrites ci-après, exprime de manière formelle le contexte introduit précédemment :

$r_1: (\textit{épaisseur}, \textit{fine}) \leftarrow (\textit{matière}, \textit{aluminium})$

$r_2: (\textit{largeur}, \textit{large}) \leftarrow (\textit{épaisseur}, \textit{fine})$

$r_3: (\textit{défaut}, \textit{mal-garnie}) \leftarrow (\textit{équipe}, \textit{novice})$

$r_4: (\textit{défaut}, \textit{mal-garnie}) \leftarrow (\textit{équipe}, \textit{intermédiaire}) \wedge (\textit{matière}, \textit{acier})$

$r_5: \textit{défaut} = \textit{cambrage} \leftarrow \textit{équipe} = \textit{experte} \wedge \textit{largeur} = \textit{large} \wedge \textit{épaisseur} = \textit{fine}$.

Certaines connaissances peuvent également décrire des faits. Ces connaissances particulières sont traduites par des règles argumentatives sans corps (par exemple, $r:(terre, ronde)\leftarrow$).

Définition 4.2.III [Fait]

Un fait est une règle argumentative r telle que $r:(a,v)\leftarrow$ avec $(a,v)\in Att(L)\times U_a$ et $body(r)=\emptyset$, $\emptyset$ désigne le fait que *body*(*r*) n'existe pas.

Un fait est une règle argumentative sans corps $r:(a,v)\leftarrow$ ou $(a,v)\in Att_\Sigma(L)\times U$ qui désigne la certitude que la valeur $v\in U$ est affectée à l'attribut $a\in Att_\Sigma(L)$.

Exemple 4.2.II [Fait]

On exprime à l'aide du fait r que la matière composant la pièce usinée est de l'aluminium.

$$r:(matière, aluminium)\leftarrow$$

Une base de connaissances Σ est constituée de faits et de règles argumentatives (cf. Définition 4.2.IV)

Définition 4.2.IV [Base de connaissances]

Une base de connaissances est un ensemble $\Sigma=R(\Sigma)\cup F(\Sigma)$ ou :

- $R(\Sigma)$ est un ensemble de règles argumentatives (cf. Définition 4.2.II) et,
- $F(\Sigma)$ est un ensemble de faits (cf. Définition 4.2.III).

Les attributs du langage L_D appartenant à la base de connaissances Σ sont dénotés par $Att_\Sigma(L_D)$.

Exemple 4.2.III [Base de connaissances]

La base de connaissances Σ d'une équipe usinant les pièces se traduit par $\Sigma=R(\Sigma)\cup F(\Sigma)$ où :

$$R(\Sigma)=\left\{\begin{array}{l} r_1: épaisseur=fine\leftarrow matière=aluminium \\ r_2: largeur=large\leftarrow épaisseur=fine \\ r_5: défaut=cambrage\leftarrow équipe=experte\wedge largeur=large\wedge épaisseur=fine \end{array}\right\}$$

$$\text{et } F(\Sigma)=\left\{\begin{array}{l} f_1: matière=aluminium\leftarrow \\ f_2: équipe=experte\leftarrow \end{array}\right\}$$

Nous définissons un argument (cf. Définition 4.2.V) de la base de connaissance Σ comme une combinaison de règles argumentatives $r\in\Sigma$ qui permettent de conclure en faveur d'un couple (*attribut*, *valeur*)$\in Att_\Sigma(L)\times U$.

Définition 4.2.V [Argument]

Soit Σ une base de connaissances. Un argument est un couple $A=(P,c)$ où c est un couple $(a,v)\in \mathsf{A}tt(\mathsf{L}_D)\times \mathsf{U}_a$ et $P\subseteq\Sigma$ est un ensemble de règles argumentatives tq

1. *P* est consistant;
2. $P\vdash c$;
3. *P* est minimal, i.e. il n'existe pas de sous-ensemble de *P* qui vérifie 1. et 2.

P est la **prémisse** de *A*, notée *premise(A)*. *c* est la **conclusion** de *A*, notée *conclusion(A)*.

On dénote par $\mathsf{A}(\Sigma)$, l'ensemble des arguments pouvant être construit à partir de Σ.

Exemple 4.2.IV [Argument]

Soit la base de connaissances Σ ,définie dans l'Définition 4.2.III, l'argument *A* tel que :

$$A=(r_1\wedge r_2\wedge r_5\wedge f_1\wedge f_2)\vdash(\textit{défaut}, \textit{cambrage})$$

permet de conclure que la large pièce composée d'aluminium et laminée par l'équipe experte présente un défaut de cambrage.

Dans la suite, les notations $A=(P,c)$ et $A=\langle c\leftarrow P\rangle$ sont sémantiquement équivalentes.

4.3 Force d'un argument

Cette section présente la notion de force que l'on associe aux arguments précédemment définis. Avant de définir formellement cette notion, on se propose de décrire la démarche sous-jacente.

4.3.1.1 Démarche

Les relations de priorité présentées dans le Chapitre 3 déterminent l'argument vainqueur en fonction de valeurs non-arbitraires établies avant la phase d'argumentation. La relation de priorité est donc « statique » et n'évolue pas durant la phase d'argumentation. Pour rendre la relation de priorité dynamique, Kakas [56][55] a proposé l'introduction d'un contexte spécifique autorisant le réordonnancement des arguments grâce à des règles contextuelles (cf. Chapitre 3). Cependant, ce contexte est limité car difficile à mettre en oeuvre sur des bases de connaissances significatives : les règles contextuelles peuvent introduire des inconsistances au au sein d'une base de connaissances, les attributs doivent être nominaux, etc.

La notion de force, introduite dans cette section, cherche à attribuer dynamiquement une force aux arguments sans avoir défini préalablement de règles contextuelles ou de quelconques

préférences sur les arguments. Pour cela, on se sert de trois composantes : 1) la fréquence des attributs, 2) la confiance de règles argumentative et 3) la structure des arguments.

1. On considère qu'un attribut est fréquent (cf. Définition 4.3.II) s'il est souvent sollicité lors de la construction d'arguments (plus exactement, dans le corps des arguments). Toutefois, nous utilisons la probabilité $P(a_i=v_i)$ avec $(a_i,v_i)\in Att_\Sigma(L)\times U_a$ pour pondérer la fréquence des attributs et éviter ainsi de maximiser leur fréquence lorsque qu'ils apparaissent dans la majorité des règles de la base de connaissances.
2. La confiance d'une règle argumentative *r* (cf. Définition 4.3.III) est calculée en fonction de la fréquence des attributs qui composent son corps et sa tête. Nous partons du principe qu'un couple $(a,v)\in Att_\Sigma(L)\times U_a$ peut être déduits d'une tête constituée principalement de faits de $F(\Sigma)$ et/ou de règles argumentatives simples et que nous pouvons avoir plus confiance en de telles prémisses.
3. Par définition, un argument est un ensemble de règles argumentative. La notion de force que nous proposons considère que les règles constituant le corps de l'argument influent sur sa force (cf . Définition 4.3.IV).

Comme la base de connaissances et les probabilités sont mises à jours durant l'argumentation, la force d'un même argument est constamment actualisée et attribuée de manière dynamique au cours de l'argumentation.

4.3.1.2 Définitions formelles

Un argument de $A(\Sigma)$ est muni d'une force dépendant à la fois de la probabilité des couples qui le composent et de la fréquence d'apparition de ces couples dans la base de connaissances Σ. La probabilité du couple $(a,v)\in Att_\Sigma(L)\times U_a$, dénotée par $P(a=v)$. Un fait $\langle (a,v) \leftarrow \rangle \in F(\Sigma)$ étant considéré comme la certitude que *a* a pour valeur *v*, nous considérons que la probabilité $P(a=v)$ est maximale.

La fréquence d'un couples $(a,v)\in Att_\Sigma(L)\times U_a$ traduit son importance au sein de $A(\Sigma)$, i.e. le fait que ce couple puisse servir à la construction d'un nombre d'arguments plus ou moins conséquent. La fréquence d'un attribut *a* pour la valeur *v* combine la probabilité et la fréquence d'apparition de (a,v) dans les règles argumentatives de Σ (cf. Définition 4.3.I)

Définition 4.3.I [Fréquence d'apparition d'un attribut]

Soient Σ une base de connaissances construite sur le langage L et $\mathsf{Att}_\Sigma(\mathsf{L})$ les attributs de Σ. La fréquence d'apparition d'un attribut $a \in \mathsf{Att}_\Sigma(\mathsf{L})$ pour la valeur $v \in \mathsf{U}_a$, notée *occurence*(a,v)

$$occurence(a,v) = \frac{|\{r \in \mathsf{R}(\Sigma) \mid (a,v) = head(r) \vee (a,v) \in body(r)\}|}{|\mathsf{R}(\Sigma)|}$$

Définition 4.3.II [Fréquence d'un attribut]

Soient Σ une base de connaissances construite sur le langage L et $\mathsf{Att}_\Sigma(\mathsf{L})$ les attributs de Σ. La fréquence d'un attribut $a \in \mathsf{Att}_\Sigma(\mathsf{L})$ pour la valeur $v \in \mathsf{U}_a$, notée frequency(a,v) est définie par :

$$frequency(a,v)=\begin{cases}1 & si\ (a,v)\in \mathsf{F}(\Sigma)\\ P(a=v)\times occurence(a,v) & sinon\end{cases}$$

Où $P(a=v)$ représente la probabilité que la valeur de a soit égale à v.

Avant de définir la force d'un argument, on introduit la notion de confiance d'une règle argumentative. La confiance d'une règle argumentative est calculée en fonction de la fréquence des couples (*attributs, valeurs*) inclus dans la tête et le corps de cette règle. Elle est la moyenne des fréquences des couples $(a,v)\in body(r)$ pondérée par la fréquence de $head(r)$.

Définition 4.3.III [Confiance d'une règle argumentative]

Soit $r\in\Sigma$ une règle argumentative. La confiance de la règle r est définie tq :

$$confidence(r)=\begin{cases}1 & si\ head(r)\in \mathsf{F}(\Sigma)\\ \dfrac{frequency(head(r))\times \sum\limits_{(a,v)\in body(r)} frequency(a,v)}{|(a,v)\in body(r)|} & sinon\end{cases}$$

La pondération par la fréquence de $head(r)$ nous permet de départager r de r' si les moyennes de $body(r)$ et $body(r')$ sont égales. En d'autres termes, on choisit la règle ayant la tête la plus pertinente.

On peut désormais, définir la force d'un argument. Un argument étant composé de règles argumentatives, on estime sa force à l'aide des confiances attribuées à ses différentes règles. Elle est définie comme suit :

Définition 4.3.IV [Force d'un argument]

Soient Σ une base de connaissances et A un argument de $\mathsf{A}(\Sigma)$. La force de l'argument A, notée $strength(A)$ est définie par :

$$strength(A)=\frac{\sum\limits_{r\in premise(A)} confidence(r)}{|premise(A)|}$$

Comme pour les logiques argumentatives (cf. Chapitre 3), la structure des arguments et leur force sont utilisées pour définir des relations de contradiction permettant de décider de leur acceptabilité.

Exemple 4.3.I [Force d'un argument]

Soit A un argument construit sur la base de connaissances Σ introduite dans l'Exemple 4.2.II :

$$A = (r_1 \wedge r_2 \wedge r_5 \wedge f_1 \wedge f_2 \wedge f_3) \vdash (défaut, cambrage)$$

la force de l'argument *A* est calculée ainsi :

$$strength(A) = \frac{\sum_{i \in \{1,2,5\}} confidence(r_i) + \sum_{i=1}^{3} confidence(f_i)}{|premise(A)|}$$

avec :

$$confidence(r_1) = P(épaisseur = fine)$$

$$confidence(r_2) = \frac{2 \times P(largeur = large) \times P(épaisseur = fine)}{3}$$

$$confidence(r_5) = \frac{P(défaut = cambrage)}{9} \times \frac{\sum_{(a,v) \in head(r_5)} frequency(a, v)}{3}$$

Les probabilités $P(a = v)$ dépendent de diverses composantes telles que l'expérience de l'équipe ou encore d'une base de cas ayant permis leur calcul.

4.4 Relations de contradiction

La base de connaissances pouvant être inconsistante, les arguments sont susceptibles d'être en conflit. Ces conflits sont mis en évidence par le biais de la relation d'attaque présentée ci-après. L'attaque est dépendante de la structure des arguments et considère que deux arguments sont effectivement en conflit, s'ils possèdent des sous-arguments contradictoires.

Définition 4.4.I [Attaque]

Soient *A* et *B* deux arguments de $\mathsf{A}(\Sigma)$. *A* attaque *B*, noté *attacks(A,B)* ssi :

$$\exists a \in \mathsf{A}tt_{\Sigma}(\mathsf{L}),\ \exists v, v' \in \mathsf{U}_a \ \ tq \ \ v \neq v',\ \exists P_1 \subseteq premise(A),\ \exists P_2 \subseteq premise(B)$$
$$\text{tels que} \ \ P_1 \vdash (a, v) et\ P_2 \vdash (a, v')$$

La relation d'attaque se base sur le fait qu'il existe des valeurs différentes dans le domaine de définition des attributs. Chaque attribut *a* ne pouvant être associée qu'à une unique valeur, si deux arguments décernent des valeurs différentes pour *a*, on dit que les arguments s'attaquent. Par abus de langage, on dira qu'un ensemble d'arguments *S* attaque *B*, noté *attacks(S,B)* ssi $\exists A \in S, attacks(A, B)$.

La relation d'attaque étant symétrique (i.e. $attacks(A, B) \Leftrightarrow attacks(B, A)$), il est impossible de déterminer le vainqueur entre les arguments *A* et *B*. C'est pour cette raison que la relation de défaite a été introduite. Lors d'une attaque, cette dernière départage les arguments grâce à leur force respective et nomme en tant que vainqueur, l'argument ayant la force la plus élevée.

Cette relation de défaite admet qu'un argument *A* défasse un argument *B*, s'ils s'attaquent mutuellement et que la force de l'argument *A* est supérieure à celle de l'argument *B*. La relation de défaite est définie comme suit :

Définition 4.4.II [Défaite]

Soient A et B deux arguments de $\mathsf{A}(\Sigma)$. *A* défait *B*, noté *defeats(A,B)* ssi :

$$attacks(A, B) \wedge strength(B) \leq strength(A) .$$

Par abus de langage, on dira qu'un ensemble d'arguments *S* défait *B*, noté *defeats(S,B)* ssi $\exists A \in S, defeats(A, B)$.

A partir de ces définition, nous pouvons définir une logique argumentative probabiliste sur la base de connaissance Σ de la manière suivante :

Définition 4.4.III [Logique argumentative probabiliste]

Une logique argumentative probabiliste est un triplet $AF = \langle \Sigma, attacks, strength \rangle$ où :
- Σ est une base de connaissances ;
- *attacks* est une relation binaire sur $\mathsf{A}(\Sigma) \times \mathsf{A}(\Sigma)$;
- *strength* est une fonction qui associe une force aux arguments de $\mathsf{A}(\Sigma)$ (cf. Définition 4.3.IV).

Par conséquent, les notions d'acceptabilité et d'admissibilité (cf. Chapitre 3) peuvent être adaptées comme suit.

Définition 4.4.IV [Libre de conflit – Acceptabilité - Admissibilité]

Soient $AF = \langle \Sigma, attacks, strength \rangle$ une logique argumentative probabiliste, $S \subseteq \mathsf{A}(\Sigma)$ un ensemble d'arguments et $A \in AF$ un argument.
1. *S* est **libre de conflits** ssi $\forall A, B \in S, \neg defeats(A,B)$;
2. *A* est **acceptable vis-à-vis** de *S* ssi :
$$conclusion(A) \in \mathsf{F}(\Sigma) \text{ ou } B \in \mathsf{A}(\Sigma), defeats(B, A) \Rightarrow defeats(S, B) .$$
3. *S* libre de conflits est **admissible** ssi $\forall B \in S$, *B* est acceptable vis-à-vis de *S*.

Afin de manipuler plus facilement l'ensemble $\mathsf{A}(\Sigma)$, nous le représentons sous la forme d'un automate appelé graphe argumentatif.

4.5 Graphe argumentatif

Un graphe argumentatif (cf. Définition 4.4.II) est une modélisation de la base de connaissances Σ. Une telle modélisation permet également la manipulation des arguments de

$A(\Sigma)$ conformément à la logique argumentative probabiliste. Afin de définir un graphe argumentatif, nous avons besoin d'introduire les ensembles AND et OR. Ces ensembles modélisent respectivement le fait qu'un attribut puisse être déduit de l'union de plusieurs attributs (i.e. l'opérateur logique $\vee$) ou de l'intersection de plusieurs attributs (i.e. l'opérateur logique $\wedge$). Formellement, un graphe argumentatif est défini comme un ensemble de noeuds représentant soit des attributs de $Att_\Sigma(L)$, soit des éléments des ensembles AND et OR . Les noeuds sont reliés entre eux par des arcs dont l'étiquette est un quadruplet (q,v,q',v') tel que q et q' sont des noeuds du graphe argumentatif et $v,v' \in U$.

Définition 4.5.I [Graphe argumentatif]

Soit Σ une base de connaissances composée de règles argumentatives. Un graphe argumentatif est un triplet $G_\Sigma = \langle Q, \sigma, \tau \rangle$ où :

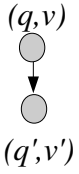
- $Q = Att_\Sigma(L) \cup OR \cup AND$ est l'ensemble des états de G avec :
  - $OR = \{or_i \mid i \in \{1,\ldots,n\}$, les noeuds représentant le ou logique$\}$ et
  - $AND = \{and_j \mid j \in \{1,\ldots,m\}$, les noeuds représentant le et logique$\}$.
- $\sigma = \{v \mid \exists a \in Att_\Sigma(L), v \in U_a\}$ est l'alphabet composé des valeurs de U_a, $a \in Att_\Sigma(L)$,
- $\tau : Q \times \sigma \to 2^Q \times 2^\sigma$ est la fonction de transition définie ci-dessous (cf. Définition 4.5.II).

Définition 4.5.II [Fonction de transition]

Soient Σ une base de connaissances et $G_\Sigma = \langle Q, \sigma, \tau \rangle$ un graphe argumentatif. La fonction de transition τ est définie comme suit :

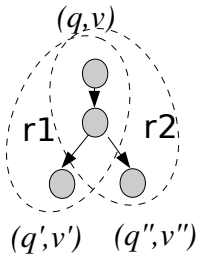
1. Si $q \in Att_\Sigma(L)$:

$$\tau(q,v) = \begin{cases} \{(q',v') \mid \exists! r \in \Sigma,\ (q',v') \subseteq body(r) \wedge head(r) = (q,v)\} \\ \{(\text{or}, v) \mid \nexists! r \in \Sigma,\ head(r) = (q,v)\} \\ \{(\text{and}, v) \mid \exists! r \in \Sigma,\ |body(r)| > 1 \wedge head(r) = (q,v)\} \end{cases}$$



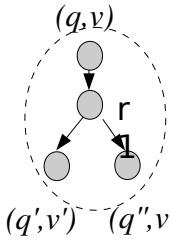
2. Si $q \in OR$:

$$\tau(q,v) = \bigcup \begin{matrix} \{\{(q',v')\} \mid \exists! r \in \Sigma,\ (q',v') \subset body(r) \wedge head(r) = (q,v)\} \\ \{\{(\text{and}, v)\} \mid \exists r \in \Sigma,\ |body(r)| > 1 \wedge head(r) = (q,v)\} \end{matrix}$$



3. Si $q \in AND$:

$$\tau(q,v) = \{(q',v') \mid \exists r \in \Sigma,\ (q',v') \subset body(r) \wedge head(r) = (q,v)\}$$



L'exemple ci-dessous met en application les définitions introduites précédemment en détaillant la fonction de transition de chacun des noeuds et en précisant la base de connaissances symbolisée par le graphe argumentatif.

Exemple 4.5.1 [Graphe argumentatif – Fonction de transition]

La Figure 4.5.2 représente un graphe argumentatif dans lequel :

La fonction de transition décrit la base de connaissance suivante :

$$\Sigma = \begin{cases} r_1 : (a_3, v_{a3}) \leftarrow (a_1, v_{a1}) \wedge (a_2, v_{a2}) \\ r_2 : (a_3, v'_{a3}) \leftarrow (a_5 = v_{a5}) \\ r_3 : (a_3, v'_{a3}) \leftarrow (a_4, v_{a4}) \end{cases}$$

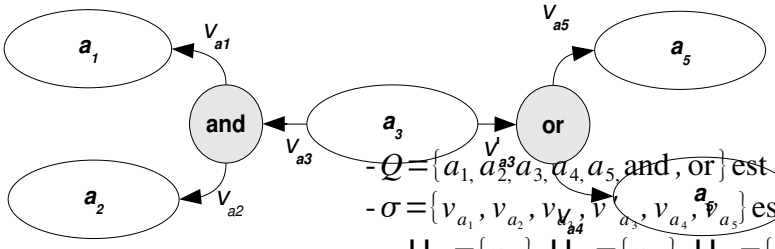


Figure 4.5.1 - Graphe argumentatif

- $Q = \{a_1, a_2, a_3, a_4, a_5, \text{and}, \text{or}\}$ est l'ensemble des états
- $\sigma = \{v_{a_1}, v_{a_2}, v_{a_3}, v'_{a_3}, v_{a_4}, v_{a_5}\}$ est l'alphabet tels que :

$$U_{a_1} = \{v_{a_1}\}, U_{a_2} = \{v_{a_2}\}, U_{a_3} = \{v_{a_3}, v'_{a_3}\}, U_{a_4} = \{v_{a_4}\}, U_{a_5} = \{v_{a_5}\}$$

- $\tau(\text{and}, v_{a_3}) = \{(a_1, v_{a_1}), (a_2, v_{a_2})\}$
- $\tau(\text{or}, v'_{a_3}) = \{(a_5, v_{a_5}), (a_4, v_{a_4})\}$
- $\tau(a_3, v_{a_3}) = \{(\text{or}, v_{a_3})\}$
- $\tau(a_3, v'_{a_3}) = \{(\text{and}, v'_{a_3})\}$

Soit un couple $(q,v) \in \mathsf{Att}_\Sigma(\mathsf{L}) \times \mathsf{U}_a$, la fonction de transition du graphe argumentatif permet de récupérer les ensembles de couples $(q',v') \in \mathsf{Att}_\Sigma(\mathsf{L}) \times \mathsf{U}_a$ apparaissant dans les règles argumentatives de Σ ayant (q,v) comme tête. Cependant, les noeuds des ensembles AND et OR nous empêchent de réaliser cette tâche directement. On utilise pour cela l'Algorithme 4.5.I.

Algorithme 4.5.I : *Premise*

ENTRÉE :

- *noeud* : un élément de $\mathsf{Att}_\Sigma(\mathsf{L})$,
- *valeur* : une valeur de l'ensemble U_{noeud}.

SORTIE :

- Les ensembles de couple $(a',v') \in \mathsf{Att}_\Sigma(\mathsf{L}) \times \mathsf{U}$ tels que $\exists r \in \Sigma, (a', v') \subseteq body(r)$.

***Premise(noeud, valeur)* :**

1 : $(q, v) := \tau(noeud, valeur)$
2 : Si $q \in \mathsf{Att}_\Sigma(\mathsf{L})$ alors
3 : Retourner $\{\{(q, v)\}\}$
4 : Si $q \in$ AND alors
5 : Retourner $\{\tau(q, v)\}$
6 : Si $q \in$ OR alors
7 : $P := \emptyset$
8 : Pour tout $(q', v') \in \tau(q, v)$ faire
9 : $P := P \cup Premise(q', v')$
10 : Fin Pour
11 : Retourner P
12 : Fin Si

L'Algorithme 4.5.I distingue trois cas, relatifs au type du noeud q atteint par $\tau(noeud, valeur)$ (cf. Définition 4.5.II) :

1. Si $q \in \mathsf{Att}_\Sigma(\mathsf{L})$, il existe une unique règle $r \in \Sigma$ telle que (*noeud, valeur*)=*head(r)* et $(q,v) \subseteq body(r)$. La tête de r se réduit alors au couple (q,v) (cf Définition 4.5.II, cas 1).
2. Si $q \in$ AND, il existe une unique règle $r \in \Sigma$ dont le corps est composé de plusieurs couples telle que (*noeud, valeur*)=*head(r)*. Par définition, la prémisse de (*noeud, valeur*) est $\{\tau(q, valeur)\}$ (cf Définition 4.5.II, cas 3).

3. Si $q \in \mathsf{OR}$, il existe plusieurs règles $r \in \Sigma$ avec (*noeud, valeur*)=*head(r)*. Chacune de ces règles peut avoir un corps composé d'un ou plusieurs couples. Dans le graphe argumentatif, cela se traduit par la présence d'arcs entre q et un ensemble de noeuds $q' \in \mathsf{A}tt_{\Sigma}(\mathsf{L}) \cup \mathsf{AND}$. La prémisse est donc l'union des prémisses des noeuds q' (cf Définition 4.5.II, cas 2).

L'Algorithme 4.5.I nous autorise à introduire une définition d'argument, basée sur les graphes argumentatifs, équivalente à la Définition 4.2.V. Soit un graphe argumentatif G_{Σ}, un argument de $\mathsf{A}(\Sigma)$ est un sous-graphe $\mathsf{G}'_{\Sigma} = \langle Q', \sigma', \tau' \rangle$ de G contenant uniquement des noeuds de $\mathsf{A}tt_{\Sigma}(\mathsf{L}) \cup \mathsf{AND}$. Mais, G' doit également contenir l'intégralité des noeuds apparaissant dans $\tau(q, v)$ ou $q \in Q' \cap Q \cap \mathsf{AND}$ et $v \in \sigma \cap \sigma'$ dans G_{Σ}. La Définition 4.5.III spécifie plus formellement cette notion d'argument.

Définition 4.5.III [Argument selon un graphe argumentatif]

Soient $\mathsf{G}_{\Sigma} = \langle Q, \sigma, \tau \rangle$ un graphe argumentatif et $\mathsf{G}'_{\Sigma} = \langle Q', \sigma', \tau' \rangle$ un sous-graphe connexe de G. G' est un argument de $\mathsf{A}(\Sigma)$ ssi :

$$\forall (a_1, v_1) \in Q' \times \sigma', \exists (a_2, v_2) \in Q \times \sigma$$

$$\tau'(a_1, v_1) = \begin{cases} \tau(\mathrm{or}, v_2) & si \ \exists \mathrm{or} \in \mathsf{OR}, \{(\mathrm{or}, v_2)\} = \tau(a_2, v_2) \\ \tau(a_2, v_2) & sinon \end{cases}$$

L'argument $A = \langle c \leftarrow P \rangle$ décrit par G'_{Σ} est défini tel que :

- $P = \bigwedge_{(a,v) \in Q' \times \sigma'} (a, v) \leftarrow Premise(a, v)$
- $c = (q, v_q) \mid \forall (a', v') \in Q' \times \sigma', \nexists \Delta \in \tau(a', v') \ tq \ (q, v_q) \in \Delta$.

On appelle racine du graphe G'_{Σ} le noeud $q \in Q'$ tel que q = *c.attribute*. Par abus de langage, on dit qu'un argument $A \in \mathsf{G}_{\Sigma}$ ssi il existe un sous-graphe G'_{Σ} représentant A.

Le graphe argumentatif, servant de représentation commune à la base de connaissances et à $\mathsf{A}(\Sigma)$, peut désormais être utilisé pour déterminer l'acceptabilité des arguments de $\mathsf{A}(\Sigma)$. Conformément à la Définition 4.4.IV, nous calculons l'acceptabilité des arguments en considérant leur force et les faits compris dans $\mathsf{F}(\Sigma)$. Pour cela, nous définissons l'algorithme suivant :

Algorithme 4.5.II : *Acceptable*

ENTRÉE :

- *node* : le noeud dont on cherche la valeur acceptable.
- *builtArguments* : les arguments construits lors du parcours.

- *markedNodes* : l'ensemble des noeuds visités.

- *acceptableArguments* : les arguments acceptables.

- Σ : une base de connaissances

SORTIE :

- La valeur acceptable *v* du noeud *noeud* ainsi que les arguments soutenant (*noeud, v*).

Acceptable(node, markedNodes, builtArguments, acceptableArguments) :

1. $temp := \langle (node, null) \leftarrow \rangle$
2. Si $node \notin markedNodes$ alors
3. $argAcceptable := \emptyset$
4. Sinon
5. $markedNodes := markedNodes \cup \{node\}$
6. Pour tout $v \in \mathsf{U}_{node}$ faire
7. Si $(node, v) \in \mathsf{F}(\Sigma)$ alors
8. $temp := \langle (node, v) \leftarrow \rangle$
9. $builtArguments(node, v) := temp$
10. $acceptableArguments := acceptableArguments \cup \{temp\}$
11. Sinon
12. Pour tout $P \in Premise(node, v)$ faire
13. $argPremise := \emptyset$
14. Pour tout $(n', v') \in P$ faire
15. *Acceptable(n', markedNodes, builtArguments, acceptableArguments)*
16. $argPremise := argPremise \cup builtArguments(n', v')$
17. Si $argPremise \neq \emptyset$ alors
18. $arg := \langle (node, v) \leftarrow Premise(node, v) \rangle$
19. $argAcceptable := argAcceptable \cup \{arg\}$
20. Si $defeats(arg, temp)$ alors
21. $argAcceptable := \{temp\}$
22. $temp := arg$
23. Si $arg.attribute = temp.attribute \wedge arg.value = temp.value$
24. Si $strength(temp) \leq strength(arg)$
25. $argAcceptable := argAcceptable \cup \{temp\}$
26. $temp := arg$
27. Fin Si

28. Fin Si
29. Fin Si
30. Fin Pour
31. Fin Pour
32. Fin Si
33. Fin Pour
34. Si $temp.value \neq null$ alors
35. $acceptableArguments := acceptableArguments \cup argAcceptable$
36. Fin Si
37. Fin Si

L'Algorithme 4.5.II détermine quelle est la valeur $v \in U_a$ la acceptable d'un attribut $a \in Att_\Sigma(L)$ (également représenté par le noeud *noeud* dans le graphe). Cette valeur est justifiée à l'aide d'arguments acceptables en faveur de (a,v). Pour cela, l'Algorithme 4.5.II effectue un parcours en profondeur du graphe argumentatif, sa complexité est de O(n).

Plus concrètement, l'Algorithme 4.5.II teste les valeurs des différents attributs. Si le couple (*attribut, valeur*) est un fait, il sera par définition acceptable. Dans le cas contraire, les noeuds de sa prémisse seront testés. L'algorithme est alors rappelé pour chacun de ces noeuds jusqu'à aboutir sur un fait. Ainsi, le sous-graphe de racine *attribut* représentant l'argument en faveur de (*attribut, valeur*) est acceptable à condition qu'il soit déduit de $F(\Sigma)$. De plus, il doit avoir une force supérieure à celle des arguments qui l'attaquent ou supporter un argument acceptable (*ligne 14* à *29*).

Le couple composé d'un graphe argumentatif et de l'algorithme d'acceptabilité est appelé graphe argumentatif probabiliste. La Définition 4.5.I introduit ce concept.

Définition 4.5.IV [Graphe argumentatif probabiliste]

> Un graphe argumentatif probabiliste est un couple $GP = \langle G_\Sigma, acceptable \rangle$ où :
> - G_Σ est un graphe argumentatif construit sur Σ (cf. Définition 4.4.II),
> - *acceptable* est l'algorithme d'acceptabilité (cf. Algorithme 4.5.II).

Ce graphe peut en réalité être assimilé à une logique argumentative. En effet, le Théorème 4.5.I prouve l'équivalence entre un tel graphe et une logique argumentative probabiliste.

Théorème 4.5.I [Théorème d'équivalence]

Un cadre d'argumentation probabiliste $AF = \langle \Sigma, attacks, strength \rangle$ est équivalent à un graphe argumentatif probabiliste $\mathsf{GP} = \langle \mathsf{G}, acceptable \rangle$.

Démonstration [Théorème d'équivalence]

La démonstration consiste à prouver que, pour une base de connaissances Σ, l'ensemble E des arguments acceptables est le même pour un cadre d'argumentation probabiliste $AF = \langle \Sigma, attacks, strength \rangle$ et un graphe argumentatif probabiliste $\mathsf{GP} = \langle \mathsf{G}_\Sigma, acceptable \rangle$. On démontre dans un premier temps que les arguments acceptables de *AF* sont également acceptables dans GP. On démontre ensuite que les arguments acceptables de GP sont acceptables dans *AF*. Soient E_{AF} et E_{GP} les ensembles d'arguments acceptables respectifs de *AF* et GP.

$\mathsf{E}_{AF} \subset \mathsf{E}_{\mathsf{GP}}$:

Soit *A* un argument de E_{AF}. Par définition, *A* est acceptable (cf. Définition 4.4.IV) :

$$\forall B \in \mathsf{A}(\Sigma), \exists E \in \mathsf{E}_{AF},\ defeats(B, A) \Rightarrow defeats(E, B) \Rightarrow attacks(E, B) \wedge strength(B) \leq strength(E)$$

D'après la Définition 4.4.I :

$$\exists a \in \mathsf{Att}_\Sigma(\mathsf{L}), \exists v, v' \in \mathsf{U}_a, \exists E' \subseteq E, \exists B' \subseteq B' \ \ tq \ \ E' \vdash (a, v) et\ B' \vdash (a, v')$$

Par conséquent, il existe dans GP un noeud $a \in Q$ tel qu'il existe deux sous-graphes $\mathsf{G}_{E'}$ et $\mathsf{G}_{B'}$ ayant comme racine commune *a*. Puisque *E'* est un sous-argument de *E*, *E'* est acceptable vis-à-vis de E_{AF}, soit la force de *E'* est supérieure à la force de *B'*, i.e. $strength(B') \leq strength(E')$ soit il existe un sous-argument de *E'* qui défait *B'*. L'algorithme d'acceptabilité permet de décider si $\mathsf{G}_{E'}$ est acceptable en effectuant chacun de ses tests (voir *l.16* et *l.22-23*).

$\mathsf{E}_{\mathsf{GP}} \subset \mathsf{E}_{AF}$:

Soit $A = \langle (a, v_A) \leftarrow P_A \rangle$ un argument de E_{GP}. D'après l'algorithme d'acceptabilité, *A* est considéré comme acceptable si :

1. La conclusion de (*a, v*) est un fait (cf. *ligne 7*) ;
2. $\forall v_B \in \mathsf{U} \mid B = \langle (a, v_B) \leftarrow P_B \rangle \in \mathsf{A}(\Sigma),\ defeats(A, B)$ et $A' \subset A$ est acceptable (cf. *ligne 13-32* avec condition *ligne 21*) ;
3. $\exists A' \in \mathsf{A}(\Sigma)\ tq\ conclusion(A) = conclusion(A')$ et *A'* est acceptable (cf. *ligne 13-32* avec condition *ligne 24*).

Par définition, *A* est également acceptable pour *AF*, i.e. $A \in E_{AF}$ et $E_{GP} = E_{AF}$.

Dans l'exemple ci-dessous est présenté le concept de graphe argumentatif, ce dernier est construit à partir d'une base de connaissances décrivant de façon brève les compétences de trois équipes de forgerons.

Exemple 4.5.II [Graphe argumentatif]

Le graphe argumentatif G décrivant la base de connaissances Σ (dont les règles sont énoncées dans l'Exemple 4.2.I) est le suivant :

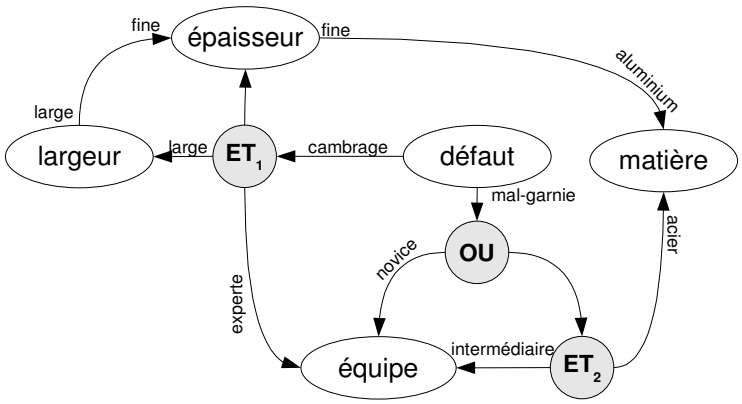


Figure 4.5.2 - Graphe Argumentatif

Le graphe argumentatif $G = \langle Q, \sigma, \tau \rangle$ est défini tel que :

$Q = \{défaut, épaisseur, équipe, largeur, matière\} \cup \{ET_1, ET_2, OU\}$;

$\sigma = \{cambrage, mal\text{-}garnie, fine, intermédiaire, novice, experte, large, aluminium, acier\}$

$\tau : Q \times \sigma \to Q \times \sigma$ est la fonction de transition définie comme suit :

- $\tau(largeur, large) = \{\{(épaisseur, fine)\}\}$
- $\tau(épaisse, fine) = \{\{(matière, aluminium)\}\}$
- $\tau(défaut, cambrage) = \{\{(équipe, experte), (largeur, large), (épaisseur, fine)\}\}$
- $\tau(défaut, mal\text{-}garnie) = \{\{(équipe, intermédiaire), (matière, acier)\} \cup \{(équipe, novice)\}\}$

4.6 Synthèse

Le cadre d'argumentation introduit ici donne la possibilité de déterminer l'acceptabilité des arguments en se basant sur leur force. Cette notion de force repose principalement sur l'expérience de l'expert et sa connaissance du domaine (cf. Définition 4.3.IV). Toutes deux peuvent être compilées dans une logique argumentative ou un graphe argumentatif afin de constituer un argumentaire propre à l'expert en question. Contrairement à [6][13][77], la force permet d'associer un ordre total, dynamique et non arbitraire aux arguments. Par conséquent, il est désormais possible de réévaluer les arguments en fonction de faits perçus, i.e. issus de l'environnement.

Bien évidemment, les graphes argumentatifs ne sont pas la seule modélisation possible des connaissances (les réseaux Bayésien sont également une solution [20][76][118]). Cependant, les graphes argumentatifs permettent de calculer l'acceptabilité des arguments selon une complexité linéaire. En effet, tout comme le parcours en profondeur cette complexité dépend uniquement du nombre d'arcs du graphe. De plus, les graphes argumentatifs permettent de réduire l'espace de recherche puisqu'ils considèrent uniquement les règles de la base de connaissances et non les dépendances causales entre les attributs de cette dernière. Enfin, les graphes argumentatifs peuvent facilement être mis à jour. La suppression (resp. l'ajout) de règles est relativement simple puisqu'il suffit de supprimer (resp. d'ajouter) des noeuds ou des arcs au graphe.

De ce fait, un argumentaire peut facilement être créé et mis à jour. Le chapitre suivant s'intéresse justement à construire un argumentaire de façon collaborative et incrémentale dans le but d'expliquer et de diagnostiquer un système.

Chapitre 5

Système multi-agent pour le diagnostic local

Sommaire

5.1 Introduction

Le diagnostic local est le résultat d'un raisonnement collectif mené par une communauté d'experts spécialisée dans un domaine précis. Les experts participent à son élaboration en échangeant un ensemble d'arguments jugés acceptables. Ces arguments sont le fruit d'un calcul d'acceptabilité réalisé à partir de leur base de connaissances et des faits inhérents au domaine. Leur base de connaissances est représentée à l'aide d'un graphe argumentatif probabiliste tel que nous l'avons introduit au chapitre précédent (cf. Chapitre 4).

Les experts réagissent aux arguments proposés soit en les enrichissant, soit en les mettant en doute. Pour cela, il est nécessaire que les experts soient autonomes et communiquent entre eux, i.e. que les experts interagissent ensemble à la construction d'un argumentaire. Ces propriétés définissent les notions de base exhibées par les agents [123][122]. C'est pour cette raison que nous modélisons une communauté d'experts à l'aide d'un système multi-agents.

Dans ce chapitre, nous définissons formellement la notion de diagnostic local (cf. section 5.3). Puis, nous proposons dans la section 5.4, l'architecture multi-agent prenant en charge l'élaboration d'un diagnostic local. Dans cette architecture, chaque expert de la communauté est représenté par un agent. Nous terminons ce chapitre en décrivant les protocoles régulant les interactions entre les agents (cf. section 5.5).

5.2 Notations

Dans cette section, nous introduisons les notations utilisées dans ce chapitre. Ces notations sont proches de celles précédemment définies (cf. Chapitre 4). C'est pour cette raison que nous faisons le parallèle entre les notations du Chapitre 4 et celles du Chapitre 5 (cf. Tableau 6).

Contrairement au Chapitre 4, un diagnostic local se base (cf. Définition 5.3.I) sur un langage restreint. Ce langage est composé d'un ensemble de termes de L appartenant à un même domaine (par exemple, la médecine ou la métallurgie). Nous dénotons par $\mathsf{D} \subset \mathsf{L}$, le langage relatif à un domaine de compétences et $\mathsf{Att}_\mathsf{D}(\mathsf{L}) \subset \mathsf{Att}(\mathsf{L})$ l'ensemble des termes composant le langage D. A l'instar des attributs, les ensembles de définitions U_D et de faits F_D sont revus pour correspondre uniquement aux attributs de $\mathsf{Att}_\mathsf{D}(\mathsf{L})$ (cf. Tableau 6). Enfin, la base de connaissance Σ_D utilisée lors de l'élaboration du diagnostic local est l'union des bases de connaissances d'une communauté $\mathcal{O} = \{ag_1, ag_2, \ldots, ag_n\}$ d'Agents du domaine (cf. Définition 5.2.I.). Nous dénotons par Σ_{ag}^{D}, la base de connaissance de l'agent $ag \in \mathcal{O}$ pour le domaine D.

Définition 5.2.I [Agent du domaine]

> Soit un domaine D, une base de connaissances Σ_{ag}^{D} et $AF = \langle \Sigma_{ag}, attacks, strength \rangle$ une logique argumentative. On appelle Agent du domaine D, une entité ag capable d'attribuer une valeur acceptable $v \in \mathsf{U}_\mathsf{D}$ à un attribut $a \in \mathsf{Att}_\mathsf{D}(\mathsf{L})$ du domaine D et d'argumenter en faveur de (a,v) grâce à la logique argumentative AF (Σ_{ag}^{D} étant la base de connaissance de ag).

	Chapitre 4	Chapitre 5
Langage	L	$\mathsf{D} \subset \mathsf{L}$
Attributs	$\mathsf{Att}(\mathsf{L})$	$\mathsf{Att}_\mathsf{D}(\mathsf{L}) \subset \mathsf{Att}(\mathsf{L})$
Ensemble de définitions	U	$\mathsf{U}_\mathsf{D} = \bigcup_{a \in \mathsf{Att}_\mathsf{D}(\mathsf{L})} \mathsf{U}_a$
Base de connaissances	Σ	$\Sigma_\mathsf{D} = \bigcup_{ag \in \mathfrak{V}} \Sigma_{ag}$ ou $\mathfrak{V} = \{ag_1, ag_2, \ldots, ag_n\}$
Faits	$\mathsf{F}(\Sigma)$	$\mathsf{F}_\mathsf{D} = \{(a, v) \in \mathsf{F}(\Sigma) \mid a \in \mathsf{Att}_\mathsf{D}(\mathsf{L})\}$

Tableau 6: - Notations du Chapitre 5

5.3 Diagnostic local

Pour un attribut $a \in \mathsf{Att}_\mathsf{D}(\mathsf{L})$ du domaine D, un diagnostic local $\Delta^\mathsf{D}(\mathsf{F}_\mathsf{D}, a)$ est un argumentaire sur a résultant d'un travail collectif, basé sur un ensemble de faits F_D, d'agents d'un même domaine D. Ce diagnostic permet de déterminer la valeur de l'attribut a acceptable par l'ensemble des agents oeuvrant sur le domaine D. Pour cela, le diagnostic local combine un ensemble de diagnostics réalisés par les agents du domaine D. Nous notons $\delta^\mathsf{D}_{ag}(\mathsf{F}_\mathsf{D}, a_j)$, le diagnostic relatif à l'attribut $a_j \in \mathsf{Att}_\mathsf{D}(\mathsf{L})$, réalisé par l'agent ag_k à partir de faits du domaine D. Ces diagnostics sont essentiels pour déterminer la valeur acceptable de l'attribut a. Chaque agent possédant sa propre base de connaissances, des inconsistances peuvent apparaître dans les arguments proposés pour justifier la valeur acceptable de l'attribut a. La réalisation des diagnostics δ^D_{ag} permet alors d'éliminer les contradictions liées à ces inconsistances en complétant l'argumentaire de Δ^D.

Définition 5.3.I [Diagnostic propre à un agent]

> Soient un agent ag appartenant au domaine D et un ensemble de faits F_D. Le diagnostic de l'attribut $a \in \mathsf{Att}_\mathsf{D}(\mathsf{L})$ propre à l'agent ag, noté $\delta^\mathsf{D}_{ag_i}(\mathsf{F}_\mathsf{D}, a)$, est défini de la manière suivante :
>
> $$\delta^\mathsf{D}_{ag_i}(\mathsf{F}_\mathsf{D}, a) = A \subseteq \mathcal{A}(\Sigma_{ag}), \exists v \in \mathsf{U}_a, (a, v) \leftarrow \mathsf{A} \cup \mathsf{F}_\mathsf{D}$$

En d'autres termes, le diagnostic propre à un agent ag est l'ensemble des arguments construits sur Σ_{ag} qui déterminent la valeur acceptable de $a \in \mathsf{Att}_\mathsf{D}(\mathsf{L})$ selon les faits F_D initialement proposés a l'agent. Grâce à la définition Définition 5.2.I, nous pouvons désormais définir un diagnostic local de façon formelle :

Définition 5.3.II [Diagnostic local]

Soient un ensemble d'agents $\mathcal{Ag}=\{ag_1, ..., ag_n\}$ et un ensemble de faits $\mathcal{F}_D$. Le diagnostic local de l'attribut $a \in Att_D(\mathcal{L})$ appartenant au domaine $\mathcal{D}$, noté $\Delta^D(\mathcal{F}_D, a)$, est défini comme suit :

$$\Delta^D(\mathcal{F}_D, a) = \bigcup_{ag \in \mathcal{Ag}, a_n \in Att_D(\Delta^D_{n-1}(\mathcal{F}_D, a_n))} \delta^D_{ag}(\mathcal{F}_D, a) \cup \bigcup_{a_{n-1} \in Att_D(\Delta^D_{n-2}(\mathcal{F}_D, a_{n-2}))} \Delta^D_{n-1}(\mathcal{F}_D, a_{n-1}) \text{ tels que}$$

$$1 \le n \le |Att_D(\mathcal{L})|, \forall k > n, \Delta^D_n(\mathcal{F}_D, a_{n-1}) = \Delta^D_k(\mathcal{F}_D, a_{k-1}) \text{ et } \Delta^D_0(\mathcal{F}_D, a) = \bigcup_{ag \in \mathcal{Ag}} \delta^D_{ag}(\mathcal{F}_D, a)$$

En résumé, le diagnostic local $\Delta^D(\mathcal{F}_D, a)$ est un ensemble stable et maximale composé des arguments permettant de justifier la valeur acceptable $v \in \mathcal{V}$ telle que $(a, v) \leftarrow \Delta^D(\mathcal{F}_D, a) \cup \mathcal{F}_D$.

Afin de réaliser ce diagnostic local, les agents ont besoin des faits relatifs à leur domaine $\mathcal{F}_D$. Les agents peuvent ainsi construire leurs arguments et débuter le diagnostic local. Les agents du domaine doivent ensuite récupérer les informations qui leur manquent pour proposer de nouveaux arguments et compléter ainsi le diagnostic local. Ces informations sont récupérées et conservées par un agent particulier, l'qgent Superviseur, dont le fonctionnement est détaillé dans la section suivante.

5.4 Architecture

Comme nous l'avons énoncé précédemment, le diagnostic local est le résultat de la collaboration d'un ensemble d'agents relatifs au domaine. Cette collaboration est orchestrée par un agent superviseur pour raisons suivante :

- La première est motivée par la réduction du nombre d'interactions et de calculs redondants. Étant donné que les qgents du domaine sont susceptibles de demander à plusieurs reprises la même information, il est préférable de la stocker en mémoire et de la mettre à jour en fonction des arguments proposés au cours du diagnostic local.
- Par ailleurs, déléguer cette tâche aux qgents du domaine peut les amener à assimiler des connaissances non pertinentes ou fortement dépendantes du cas traité, i.e. de l'ensemble des faits.

Pour un domaine, notre système multi-agent se compose donc d'une communauté d'agents du domaine (cf. section 5.4.1) et d'un agent superviseur (cf. section 5.4.2). Nous précisons dans les sections 5.4.1 Erreur : source de la référence non trouvée et 5.4.2, le comportement des agents de la communauté $\mathcal{Ag}$ lors de la construction du diagnostic local.

5.4.1 Agent du domaine

À chaque agent du domaine est rattaché un graphe argumentatif probabiliste $\mathcal{G}$ synthétisant les règles argumentatives présentes dans sa base de connaissances Σ. L'agent ag_i (cf. Définition 5.2.I) doit, en fonction des faits $\mathcal{F}_D$ qu'on lui présente, construire un argumentaire en faveur d'un attribut $a \in \mathrm{Att}_D(\mathcal{L})$. Le but de cet argumentaire est de trouver grâce aux règles argumentatives de Σ, une valeur $v \in \mathcal{U}$ jugée acceptable par ag_i. Cet argumentaire est uniquement composé d'arguments acceptables de $\mathcal{G}$ soutenant (a,v). Ensemble, ces arguments décrivent le diagnostic $\delta^{D}_{ag_1}(\mathcal{F}_D, a)$ propre à l'agent ag_i.

Toutefois, l'agent peut avoir besoin d'informations supplémentaires pour établir son diagnostic. Les informations demandées sont les valeurs attribuées à certains attributs de $\mathrm{Att}_\Sigma(\mathcal{L})$. Dans ce cas, il lui est possible d'interroger d'autres agents du domaine pour les obtenir. Afin d'éviter des calculs redondants, l'agent Superviseur (cf. section 5.4.2) sert d'intermédiaire. Différentes stratégies peuvent être adoptées par l'agent du domaine suivant l'information dont il a besoin :

1. *Souscrire des demande d'information* relative à un attribut $a \in \mathrm{Att}_\Sigma(\mathcal{L})$ auprès de l'agent superviseur : obtenir la valeur de l'attribut a et être notifié lorsque sa valeur est modifiée. Les agents sont contraints de s'abonner pour tout attribut leur permettant de compléter leur diagnostic $\delta^{D}_{ag_1}(\mathcal{F}_D, a)$.
2. *Demander d'information temporaire :* initiée par l'agent lorsqu'aucune information sur les attributs souscrits n'est disponible. De cette façon, l'agent du domaine a la possibilité de construire un diagnostic partiel apportant tout de même des nouvelles informations aux autres Agents du domaine.

La Figure 5.4.1 détaille comment l'agent du domaine construit son diagnostic. Nous nous proposons d'expliquer ce processus. Pour cela, nous supposons que l'agent du domaine a déjà été informé des faits F_D. Le diagnostic porte sur l'attribut $a \in Att_D(L)$.

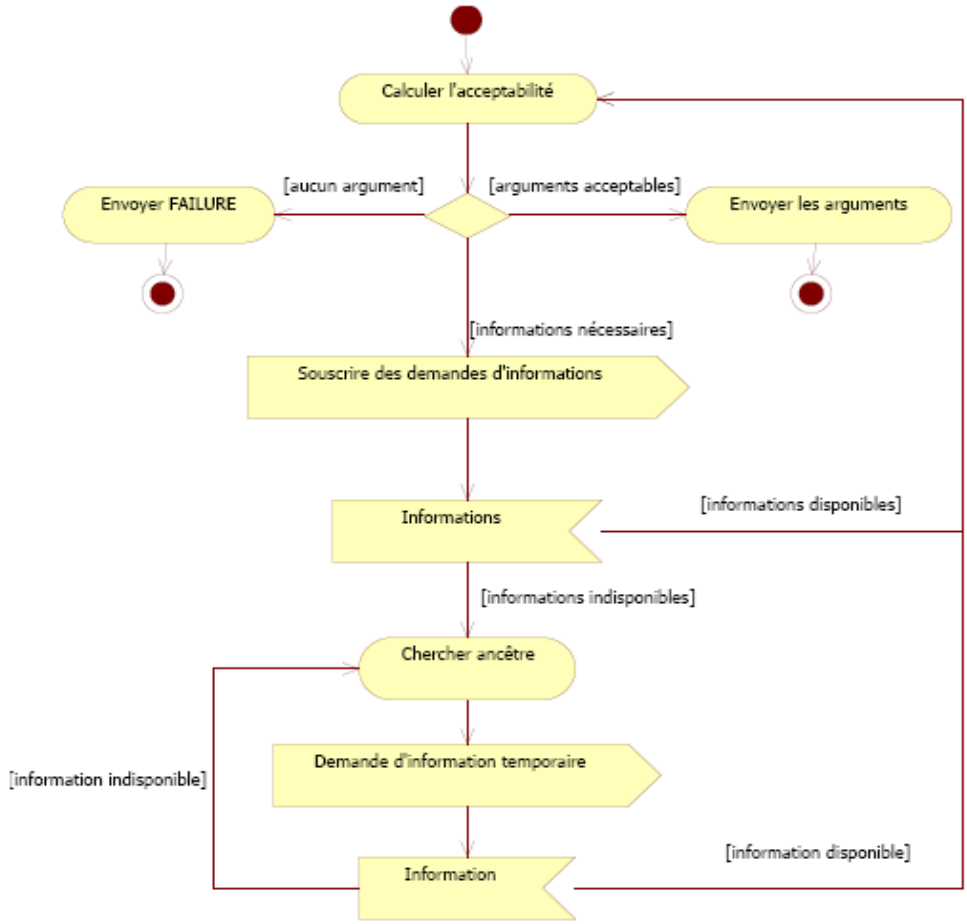


Figure 5.4.1: - Comportement d'un Agent du domaine

Dans un premier temps, l'agent construit un diagnostic $\delta^{D}_{ag_1}(F_D, a)$ (*Calculer* l'acceptabilité) en faveur de *a* basé uniquement sur les faits. Ce diagnostic est transmis à l'Agent Superviseur qui pourra ainsi fournir de nouvelles informations aux autres Agents du domaine ([*arguments acceptable*]). Si l'agent est incapable de construire un tel argumentaire ([*aucun argument*]), il prévient l'agent Supervieur (*Envoyer FAILURE*) et souscrit des demandes d'information ([*informations nécessaires*]).

1. La construction du diagnostic, même partielle, permet à l'agent de détecter les informations dont il a besoin pour compléter son argumentaire. Ainsi, il souscrit une demande d'information pour chacune d'elle (*Souscrire des demande d'information*).
2. Si aucune information n'est disponible ([*Informations indisponibles*]), l'agent cherche alors les attributs pouvant être déduits des attributs $a' \in Att_D(L)$ (tq.

$\exists r \in \delta^{D}_{ag_1}(F_D, a), (a', v) \in head(r)$ et $v \in U$) ayant fait l'objet d'une demande d'information permanente ou temporaire (*Chercher ancêtre*). Toute information ainsi obtenue ([*Information disponible*]) permet à l'agent du domaine de peaufiner son diagnostic.

Comme on a pu le constater, le rôle de l'agent Superviseur est important. En plus d'être l'intermédiaire des agents du domaine, on lui dédie la charge de calculer le diagnostic local $\Delta^{D}(F_D, a)$. Pour cela, il doit construire et justifier son argumentaire avec l'aide des Agents du domaine. Dans la section suivante, nous introduisons l'agent Superviseur et présentons en détail son fonctionnement.

5.4.2 Agent Superviseur

La principale tâche de l'agent Superviseur est l'élaboration du diagnostic local $\Delta^{D}(F_D, a)$ de l'attribut $a \in Att_{\Sigma}(L)$. Dans la suite, nous dirons que a est le thème du diagnostic. Le comportement réalisant cette tâche est représenté par la Figure 5.4.2. L'agent Superviseur ne possédant aucune connaissance sur le domaine D, il s'appuie sur les diagnostics des Agents du domaine. Dans ce but, il transmet aux agents du domaine les faits F_D. Si nécessaire, les faits sont actualisés au début de l'élaboration du diagnostic local. Le diagnostic local porte sur un attribut particulier de l'individu et sert à justifier la valeur qui lui a été attribuée.

Le thème est soumis aux agents du domaine (*Transmettre le thème*) qui, une fois calculé, communiquent leur diagnostic à l'agent Superviseur. Les arguments ainsi reçus permettent de construire la base de l'argumentaire de l'agent Superviseur. Cet argumentaire est représenté à l'aide d'un graphe argumentatif probabiliste G_S (*Mise à jour du graphe argumentatif*).

Bien que ces arguments soient construits à partir de faits, ils sont composés de règles argumentatives dont les couples (*attribut, valeur*)$\in Att_{\Sigma}(L) \times U$ peuvent être remis en question. Les règles étant généralement propres à un agent du domaine, la valeur du couple (*attribut, valeur*) est possiblement non acceptable par d'autres Agents du domaine. C'est pour cette raison que l'Agent Superviseur requiert, pour chacun de ces attributs, l'avis des agents du domaine (*Transmettre les sous-thèmes*). À leur réception, les arguments justifiant ces sous-thèmes sont ajoutés à G_S. L'argumentaire de l'agent Superviseur est ainsi complété tant qu'il existe des sous-thèmes non justifiés dans $\Delta^{D}(F_D, a), a \in Q$ l'ensemble des états de G_S.

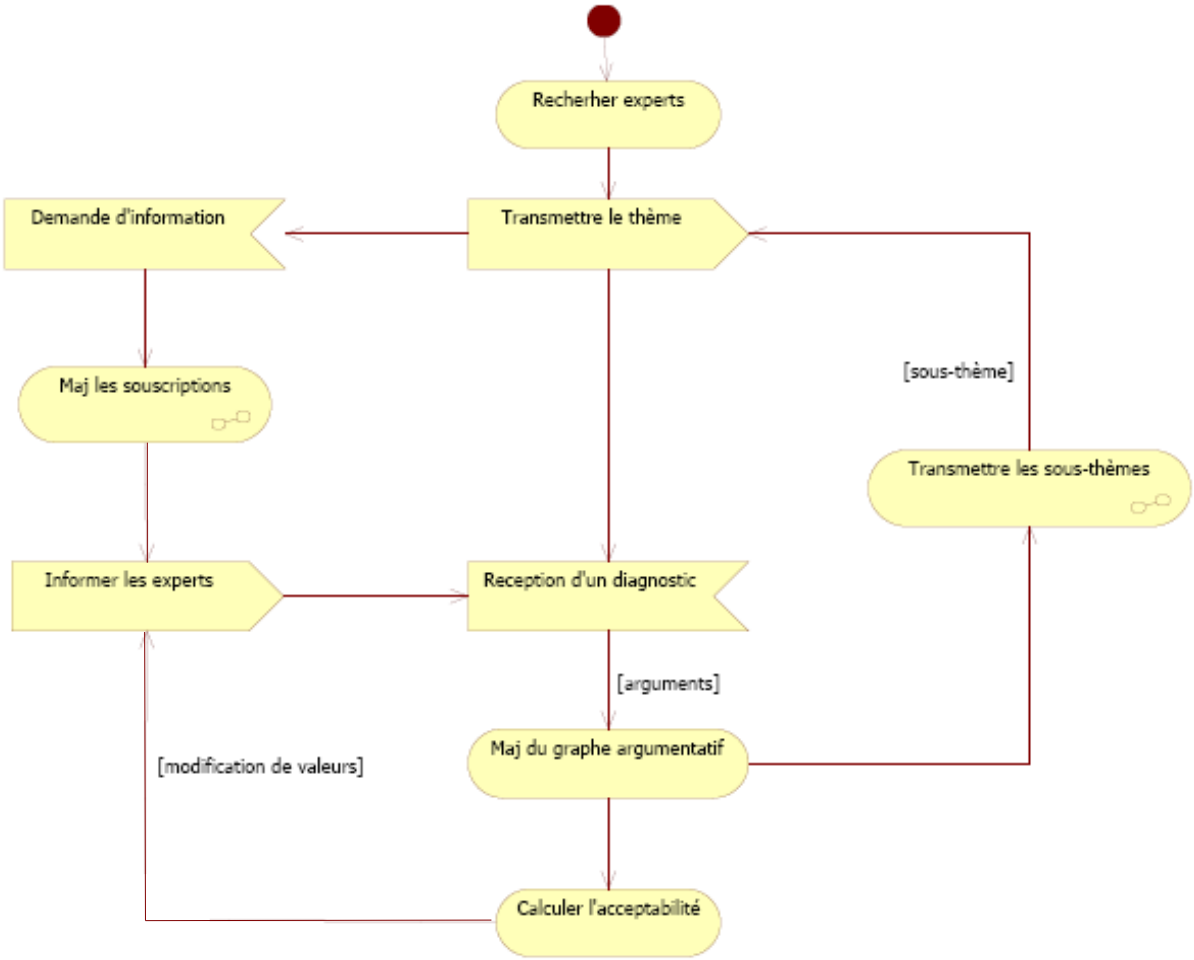


Figure 5.4.2 - Comportement de l'Agent Superviseur

Dans la section précédente (cf. section 5.4.2Erreur : source de la référence non trouvée), nous avons énoncé que l'agent Superviseur servait d'intermédiaire entre les agents du domaine. Ce dernier traite les demandes différemment suivant que l'attribut souscrit appartienne ou non à Δ^{D}, i.e. à son graphe argumentatif G_S :

1. Si $a \in \Delta^{D}(F_D, a)$, l'agent Superviseur communique la valeur v considérée acceptable pour a. Cette valeur est estimée à l'aide de l'algorithme d'acceptabilité (cf. Chapitre 4) utilisé sur le graphe argumentatif de G_S (Calculer l'acceptabilité).
2. Si $a \notin \Delta^{D}(F_D, a)$, l'agent Superviseur élabore le diagnostic local $\Delta^{D}(F_D, a)$ de l'attribut a. Ce diagnostic local est réalisé avec l'aide des Agents du domaine comme décrit précédemment. Ensuite, l'agent Superviseur communique la valeur acceptable de a calculée comme en 1.

Les informations sont ensuite transmises aux agents du domaine (*Informer les experts*). Les nouveaux arguments construits à l'aide de ses informations viennent compléter l'argumentaire de l'agent Superviseur.

Dans le cas d'une souscription, l'agent Superviseur informe les agents du domaine des mises à jour effectuées sur les valeurs des attributs qu'ils ont souscrits, i.e. après un calcul d'acceptabilité (*Calculer l'acceptabilité*). Pour cela, l'agent Superviseur se réfère à sa liste des souscriptions qui relie les agents du domaine aux attributs. Cette liste comprend chacune des souscriptions permanentes requises par les agents du domaine (*Maj les souscriptions*).

Afin de communiquer ces informations et créer le diagnostic local, il est nécessaire que les agents du domaine et l'agent Superviseur interagissent. Ces interactions sont définies dans la section suivante.

5.5 Protocoles de communication

Dans cette section, nous proposons de détailler les interactions permettant aux agents du domaine D de réaliser les tâches qui leur sont attribuées. Ces interactions sont modélisées à l'aide de protocoles décrits en AUML (Agent UML)[8] et respectent les spécifications de la FIPA (Foundation for Intelligent Physical Agent)[33][34][35][36].

Trois protocoles ont été identifiés, chacun exprime un besoin de communication de l'agent pour la réalisation du diagnostic local. Ainsi, les tâches pour lesquelles une interaction est nécessaire sont : la recherche d'informations, les demandes d'informations temporaires et les souscriptions permanentes (cf. section 5.4.1Erreur : source de la référence non trouvée et 5.4.2). Les protocoles utilisés à cet effet sont respectivement présentés dans les sections 5.5.1, 5.5.2 et 5.5.3.

5.5.1 Protocole de recherche d'informations

Le protocole de recherche d'informations (cf. Figure 5.5.1) décrit comment est élaboré le diagnostic local Δ^D mais également comment il est complété suite aux demandes des Agents du domaine (cf. section Erreur : source de la référence non trouvée5.3). À la demande de l'agent Superviseur, représentée par un message *Request*, les agents du domaine construisent un argumentaire en faveur d'un ou plusieurs attributs $a \in Att_D(L)$. Ils utilisent pour cela l'algorithme d'acceptabilité associé à leur graphe argumentatif probabiliste et l'ensemble des faits F_D.
Suivant le résultat fourni par l'algorithme d'acceptabilité, l'agent ag_i peut :

1. Signaler à l'agent Superviseur qu'il ne possède aucune connaissance sur l'attribut *a* en envoyant un message ayant comme performatif *Not_understood*.

2. Informer l'agent Superviseur des arguments acceptables qu'il a pu construire, i.e. les arguments constituant $\delta^{D}_{ag_i}(F_D, a)$. Pour cela, il transmet un message *Inform* dont le contenu est les arguments de $\delta^{D}_{ag_i}(F_D, a)$.
3. Informer l'agent Superviseur qu'il est incapable de calculer son diagnostic en raison d'un manque d'informations. A cet effet, l'agent ag_i envoie un message *Failure* en précisant dans le contenu l'attribut pour lequel le diagnostic n'a pas pu être calculé.

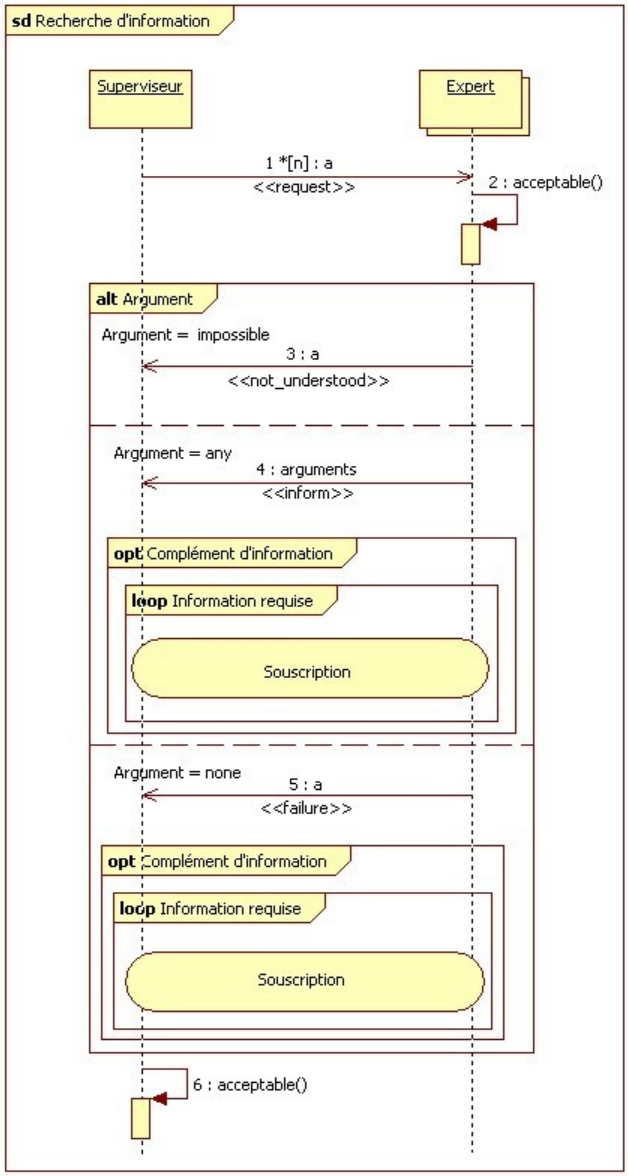


Figure 5.5.1 - Protocole de recherche d'informations

Dans les cas 2. et 3., l'agent ag_i va souscrire auprès de l'agent Superviseur un complément d'information de façon permanente ayant pour but de l'aider à peaufiner son diagnostic (cf.

section Erreur : source de la référence non trouvée). Le protocole de souscription est détaillé dans la section 5.5.3. Pour terminer, l'agent Superviseur calcule l'acceptabilité des attributs $a \in \mathsf{Att}_D(\mathsf{L})$ pour lesquels il a demandé un diagnostic. L'acceptabilité de ces attributs n'est pas définitive car elle est soumise à des arguments futurs construits à l'aide des informations souscrites. Néanmoins, elle permet de découvrir les thèmes nécessaires pour le faire (cf. section 5.4.2).

Avant de décrire le protocole de souscription, nous allons présenter le protocole de demandes d'informations temporaires. Ce protocole permet aux agents du domaine d'obtenir des informations dans le cas ou les informations souscrites ne sont pas disponibles.

5.5.2 Protocole de demande d'informations temporaires

Le protocole de demandes d'informations temporaires (cf. Figure 5.5.2) sert également à pallier le manque d'informations des agents du domaine (cf. section Erreur : source de la référence non trouvée). Pour cela, l'agent du domaine fait une demande auprès de l'agent Superviseur. Cette demande se présente sous la forme d'un message *Query_ref* ayant pour contenu l'attribut $a \in \mathsf{Att}_\Sigma(\mathsf{L})$ dont la valeur acceptable $v \in \mathsf{U}$ est souhaitée.

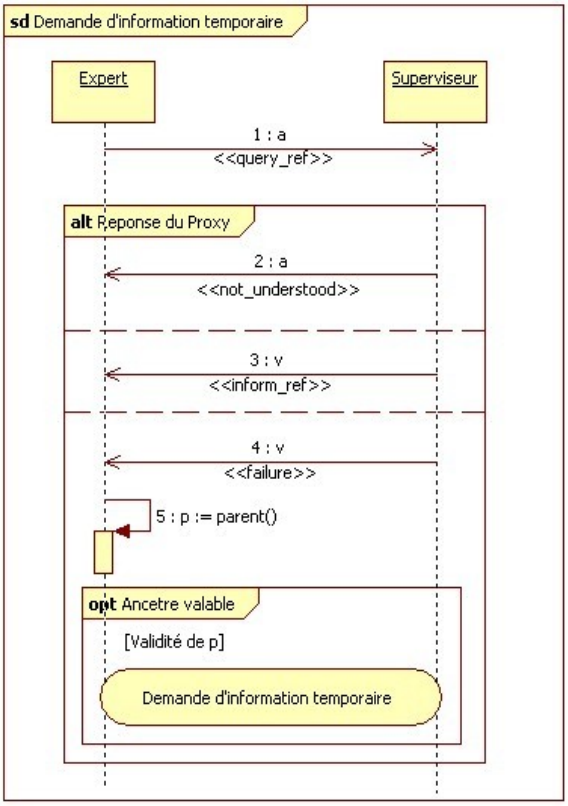


Figure 5.5.2 - Protocole de demande d'information temporaire

L'Agent Superviseur suivant ses connaissances peut :

1. Signaler à l'agent du domaine que $a \notin Att_D(L)$ par l'envoi d'un message *Not_understood* dont le contenu est l'attribut *a*.
2. Informer l'agent du domaine de la valeur qu'il juge acceptable lors de la réception de la demande en répondant par un message *Inform_ref* contenant la valeur *v*.
3. Signaler à l'agent du domaine qu'il est dans l'incapacité de déduire une valeur acceptable pour l'attribut *a* (message n°4). Suite au message *Failure*, l'agent du domaine a la possibilité de faire une nouvelle demande pour un ancêtre de *a* (cf. section Erreur : source de la référence non trouvée).

Nous sommes désormais capables de présenter le protocole de souscription qui se base sur les protocoles de recherche d'informations et de demande d'informations.

5.5.3 Protocole de souscription

Le protocole de souscription (cf. Figure 5.5.3) décrit formellement comment interagissent les agents du domaine avec l'agent Superviseur lors d'une souscription permanente (cf. section

Erreur : source de la référence non trouvée). La demande est représentée par un message *Subscribe* ayant pour contenu l'attribut $a \in \mathrm{Att}_{\Sigma}(\mathsf{L})$. Si l'agent Superviseur possède l'information, un message *Inform* contenant la valeur acceptable $v \in \mathsf{U}$ de a fait suite à cette demande.

Dans le cas contraire, l'agent Superviseur initie une recherche d'informations auprès des Agents du domaine de $\mathfrak{V}$. Au terme de cette recherche d'informations, il est capable de calculer l'acceptabilité de l'attribut a. Deux possibilités peuvent alors se présenter :

1. L'agent Superviseur a pu construire un diagnostic local $\Delta^{D}(F_{D}, a)$ en faveur d'une valeur $v \in U_a$ appartenant à l'ensemble de définition de *a*. Il transmet alors cette valeur à l'agent du domaine à l'aide d'un message *Inform*.
2. L'agent Superviseur a été dans l'incapacité de construire un argumentaire (*Réception d'information extérieure*) en faveur d'une quelconque valeur de l'ensemble de définition U_a. Il prévient alors l'agent du domaine (message n°5) qui pourra faire une demande d'information temporaire. Dans le cas ou l'agent du domaine reçoit une information de la part de l'agent Superviseur (*Acceptabilité*), il recalcule l'acceptabilité de *a*. Puis notifie l'agent Superviseur du résultat.

Σ_1	Σ_2
R_{11} : (sport, non) $\Rightarrow$ (cardiaque, oui)	R_{21} : (surpoids, oui) $\Rightarrow$ (poids, élevé) $\wedge$ (taille, petite)
R_{12} : (cardiaque, oui) $\Rightarrow$ (age, 60+) $\wedge$ (surpoids, oui)	R_{22} : (sport, oui) $\Rightarrow$ (surpoids, oui)
	R_{32} : poids=élevé $\Rightarrow$ age=60+

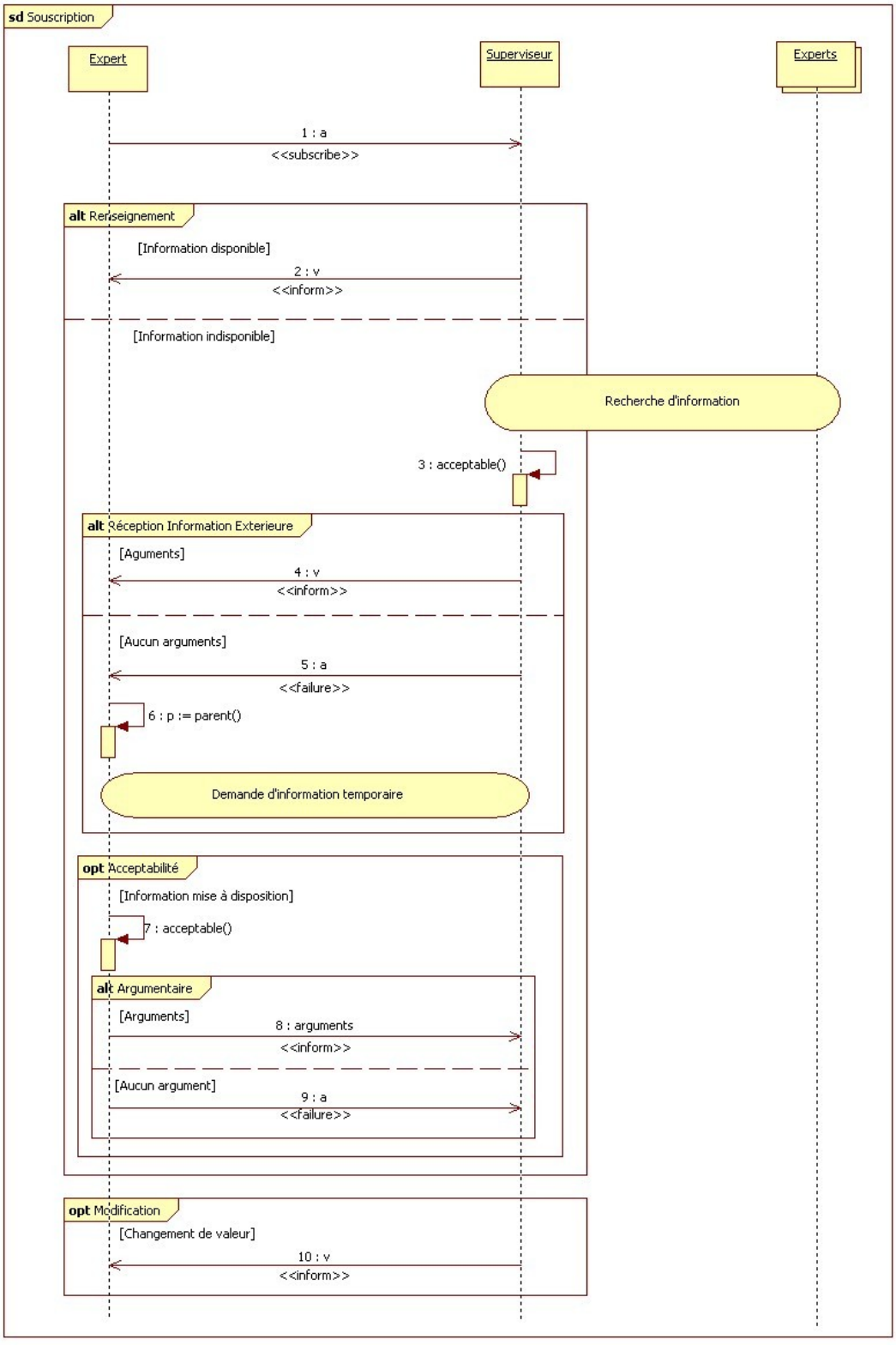


Figure 5.5.3 - Protocole de souscription

L'agent Ag_2 fournit à l'agent Superviseur , l'argument suivant : $S=\langle(surpoids\,,oui),\{R_{22},R_{32},(taille\,,petite),(\text{age, 60+})\}\rangle$. Comme aucun argument contredit *S*, ce dernier considère que '*oui*' est la valeur acceptable de *surpoids* et transmet cette valeur à l'agent Ag_1. Désormais, l'agent Ag_1 a la possibilité de construire son diagnostic. Celui-ci est décrit à l'aide de l'argument $C=\langle(sport\,,non),\{R_{11},R_{12},(taille\,,petite),(surpoids\,,oui)\}\rangle$ dont la force, attribuée par Ag_1 est égale à 0,8.

Lorsque l'agent Superviseur juge qu'une valeur *v* de U_a est plus acceptable que *v'*, il alerte l'agent du domaine de la nouvelle valeur de *a* (message n°10).

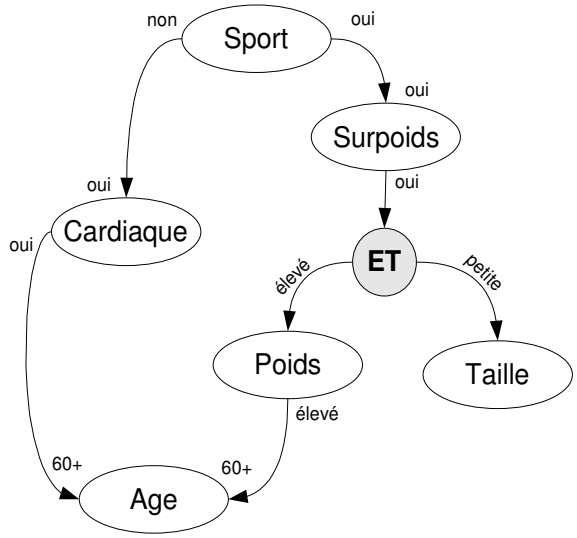


Figure 5.6.1 - Argumentaire de l'Agent Superviseur

5.6 Exemple

Dans cette section, nous schématisons la création d'un diagnostic local par une communauté de trois agents : un agent Superviseur et deux agents du domaine (Ag_1 et Ag_2). Les agents du domaine sont tous deux médecins, leur base de connaissances respective sont représentées par Σ_1 et Σ_2.

Les agents du domaine ne peuvent plus fournir d'arguments supplémentaires en faveur du *sport* ou d'autres attributs (*cardiaque*, *surpoids*, *poids*). L'agent Superviseur considère que l'argumentaire est complet, i.e. qu'il n'existe plus d'argument attaquant *C* et *P* (cf. Définition Attaque). Le diagnostic local $\Delta^{D}(\Sigma_D, sport)$, représenté par le graphe argumentatif de la Figure 5.6.1, permet à l'agent Superviseur de déterminer la valeur acceptable de l'attribut *sport*. Cette dernière est la conclusion de l'argument ayant la force la plus élevée, i.e. la conclusion de *C*. Par conséquent, on déconseille la pratique d'un sport à Robert.

5.7 Synthèse

Dans ce chapitre, nous avons pu voir qu'il était possible pour une communauté d'agents d'établir un diagnostic collectivement. Un tel diagnostic, appelé diagnostic local, est le résultat d'un dialogue entre agent Superviseur et agents du domaine. Lors de ce dialogue, des arguments jugés acceptables par les Agents du domaine sont échangés. Ils représentent l'opinion des agents du domaine et sont établis sur leurs bases de connaissances et les faits décrivant l'individu ou le système à diagnostiquer.

La centralisation du diagnostic par l'agent Superviseur est justifiée. Les demandes d'informations étant traitées par l'agent Superviseur, leur résultat est disponible à tout moment auprès de ce dernier. Les agents du domaine ne sont pas contactés évitant ainsi qu'ils calculent l'acceptabilité de l'attribut demandé. Par conséquent, la centralisation permet d'éviter des redondances dans les calculs et les interactions, critère pouvant influer grandement sur les performances du système. Par ailleurs, on souhaite garder l'identité des agents du domaine. Cette identité, symbolisée par la base de connaissances de l'agent du domaine, reflète son expérience. Cette expérience, décrite par un ensemble de règles, est représentative de la base de cas de l'expert. Or, l'assimilation des arguments proposés par d'autre agents influence le raisonnement de l'agent du domaine qui établit alors un diagnostic en s'appuyant sur des règles ne respectant pas forcément sa base de cas. Dans ce cas, la centralisation permet de considérer les conclusions d'autres agents du domaine sans altérer la base de connaissances d'un agent du domaine, i.e. son identité.

Cependant, diagnostiquer un problème requiert souvent des connaissances relatives à plusieurs domaines. L'obésité, par exemple, peut être engendrée par une mauvaise alimentation, un problème psychologique, un dysfonctionnement génétique ou impliquée par plusieurs de ces facteurs. En général, les experts sont spécialisés dans un unique domaine. Un nutritionniste ne possède aucune connaissance relative à la psychologie. Bien qu'un attribut puisse apparaître dans plusieurs domaines (i.e. $Att_{D_1} \cap Att_{D_2} \neq \emptyset$), les arguments sont majoritairement fondés sur des attributs du domaine auquel appartient l'agent du domaine. Ainsi, lorsque des agents du domaine de domaines différents s'échangent des arguments, il leur est difficile de les mettre en doute ou de les défendre.

C'est pour cette raison que nous introduisons entre les domaines un média de communication. Ce média lie les domaines entre eux et permet d'élaborer un diagnostic sur l'ensemble des domaines concernés par le problème. L'élaboration d'un tel diagnostic, dit global est présenté dans le chapitre suivant.

Chapitre 6

Diagnostic global

Sommaire

6.1 Introduction

Le problème que nous souhaitons diagnostiquer est pluridisciplinaire. Les décisions prises par un agent du domaine *A* sont susceptibles d'impacter le domaine *B* ou inversement. Cependant, les langages des domaines *A* et *B* sont caractérisés par des attributs différents. Par conséquent, nous devons mettre en place un média permettant aux Agents de collaborer. Pour cela, nous avons opté pour un modèle statistique, résultant d'un apprentissage sur les bases de cas utilisées par les agents : un réseau Bayésien.

Dans ce chapitre, nous commençons par introduire la notion de diagnostic global en présentant les processus utilisés pour i) la construction du réseau Bayésien (cf. section 6.2.1) et ii) l'intégration des diagnostics locaux au sein de ce modèle statistique (cf. section 6.2.2). Ce

réseau Bayésien est ensuite géré par un agent particulier, l'agent Bayésien, dont nous présentons le comportement dans la section 6.3.1 et la section 6.3.2. L'agent Bayésien étant désormais introduit, nous donnons en section 6.3.3, une vision détaillée de l'architecture de notre système multi-agents.

6.2 Diagnostic globalErreur : source de la référence non trouvée

Le diagnostic global permet, grâce à un modèle statistique, de prendre en considération les connaissances des experts, représentés dans le système par les différents agents du domaine. Le modèle statistique considéré est un modèle probabiliste nommé réseau Bayésien. Un tel modèle permet non seulement de représenter la connaissance, mais également d'associer une probabilité à un ensemble de faits (décrit dans le réseau). Cette probabilité est inférée à partir des relations causales modélisées par le réseau et de certains faits.

Les connaissances d'un expert étant liées à un unique domaine, les dépendances causales ne peuvent être construites autrement que par apprentissage. En effet, il faudrait que les experts puissent construire et évaluer des règles concernant l'ensemble de domaines, tâche compliquée lorsque les experts ne possèdent des connaissances propres qu'à leur domaine de compétence. Dans la section 6.2.1, nous décrivons le processus utilisé pour obtenir l'ensemble d'apprentissage sur lequel le réseau Bayésien est appris. Nous décrivons ensuite (cf. section 6.2.2) comment le diagnostic global est élaboré, nous utilisons pour cela les propriétés inhérentes des réseaux Bayésiens.

6.2.1 Réseau Bayésien

Les réseaux Bayésiens ont pour objectif d'acquérir, représenter et utiliser la connaissance (cf. Chapitre 1). Ces derniers décrivent, sous forme de graphes orientés acycliques, les dépendances causales existant à l'intérieur d'un ensemble de variables. Ces dépendances sont probabilisées et permettent d'attribuer une probabilité $P(a=v)$ au couple (a,v), a étant une variable et v une valeur de l'ensemble de définition de a. Ces probabilités sont calculées à l'aide de la formule de Bayes et nécessitent les tables de probabilités conditionnelles associées aux variables. Ces tables répertorient les probabilités conditionnelles d'une variable en fonction des valeurs possibles de ses variables parentes. On appelle $Parent(a)$ l'ensemble des variables parentes de a, i.e. les causes dont a dépend directement. Formellement, la probabilité conjointe des variables du réseau est définie comme le produit des probabilités locales de chaque variable en fonction de ses parents :

$$P(a_{1,},\ldots,a_n)=\prod_{i=1}^{n} P(a_i \mid Parent(a_i))$$

6.2 DIAGNOSTIC GLOBALERREUR : SOURCE DE LA

L'intérêt des réseaux Bayésiens réside dans le fait qu'ils considèrent non seulement les connaissances a priori des agents, mais également l'expérience contenue dans les données. Dans notre cas, les problèmes considérés font intervenir plusieurs domaines d'expertise. Généralement, les experts possèdent des connaissances uniquement sur leur domaine d'expertise. Il est alors difficile de construire un réseau Bayésien permettant de mettre en exergue les dépendances inter-domaines uniquement à l'aide des connaissances des experts. C'est pour cette raison que nous utilisons les bases de cas des experts comme ensemble d'apprentissage. Dans la suite, nous nous focalisons sur la construction de cet ensemble d'apprentissage. Les personnes souhaitant de plus amples informations sur les réseaux Bayésien et l'apprentissage de leur structure pourront consulter les références suivantes **[TODO_REF]**. Pour construire le réseau Bayésien, nous faisons les trois hypothèses suivantes [11], [21] et [84].

1. Les cas relatifs à un agent ag_1 du domaine D_i sont décrits à l'aide de tous les attributs $a \in Att_{D_i}$ caractérisant le domaine D_i. Si $Att_{ag_i} \subset Att_{D_i}$ cela signifie que seulement une partie des attributs de D_i sont pertinents selon ag_i pour construire sa base de connaissance.

1. Il existe un attribut *Id*, commun à l'ensemble des domaines, permettant de d'identifier les données relatives à un cas dans les différents domaines.

2. Les cas présents dans le système ont été observés par au moins un agent de chaque domaine intervenant dans la résolution du problème. Le cas est alors caractérisé dans la base de cas de ag_1, notée σ_{ag1}, par un ensemble de couples (a_i,v) tels que : $\forall j \in \{1,\ldots,n\}, \forall a_i \in Att_{D_j} \cup \{Id\}, \exists v \in U_{a_i} \mid (a_i, v) \in \sigma_{ag_1}$.

Les différents couples (*attribut, valeur*) appartenant aux domaines D_i sont fusionnés suivant l'attribut *Id*. Le résultat de cette fusion permet de regrouper les données d'un même individu constituant ainsi l'ensemble d'apprentissage. L'Exemple 6.2.I schématise la construction de l'ensemble d'apprentissage. Cet ensemble d'apprentissage est en partie utilisé pour apprendre la structure et les tables de probabilités conditionnelles de notre réseau Bayésien. La partie restante sert ensuite à valider le réseau par cross-validation (doc).

Ce réseau est alors utilisé pour diagnostiquer les nouveaux individus. On utilise pour cela, les données relatives à cet individu et certaines propriétés inhérentes aux réseaux Bayésiens. Le processus de diagnostic est expliqué dans la section suivante.

Exemple 6.2.I [Réseau Bayésien]

On cherche à construire un réseau Bayésien par rapport aux connaissances de trois experts : un généticien, un nutritionniste et un sociologue. Ce réseau Bayésien est utilisé pour déceler les origines et la sévérité du niveau d'obésité d'une personne.

Ainsi, le généticien se base sur l'héritage génétique de la personne. En effet, l'obésité est une maladie pouvant être héritée génétiquement des parents. Une personne ayant des parents obèses aura tendance à être elle-même en surpoids. Le nutritionniste, quant à lui, s'appuie sur l'alimentation de la personne. Une alimentation riche en lipides ou glucides est peu saine. Au contraire, une alimentation équilibrée est caractérisée par une consommation égale en protides, glucides et lipides. Enfin, le sociologue se base sur l'activité de la personne. Une personne ayant un travail de bureau et peu sportive est considérée comme sédentaire. Par conséquent, elle a besoin d'un apport calorique moindre.

Id	Alimentation	Lipide	Protide	Glucide
1	Équilibrée	Raisonnable	Raisonnable	Raisonnable
2	Équilibrée	Raisonnable	Raisonnable	Raisonnable
3	Équilibrée	Raisonnable	Raisonnable	Raisonnable
4	Non équilibrée	Excès	Insuffisant	Excès
5	Équilibrée	Raisonnable	Raisonnable	Raisonnable
6	Non équilibrée	Excès	Excès	Raisonnable

Id	Gène	Gène_mère	Gène_père
1	Imc20-25	Imc20-25	Imc30+
2	Imc30+	Imc30+	Imc30+
3	Imc20-25	Imc20-25	Imc20-25
4	Imc30+	Imc30+	Imc25-30
5	Imc20-25	Imc30+	Imc20-25
6	Imc30+	Imc25-30	Imc30+

Id	Sédentaire	Travail	Sportif
1	Non	Manutention	Non
2	Oui	Cadre	Oui
3	Oui	Cadre	Oui
4	Non	Cadre	Non
5	Non	Manutention	Oui
6	Oui	Cadre	Non

Fusion des bases de cas selon Id

Id	Gène	Gène_mère	Gène_père	Sédentaire	Travail	Sportif	Alimentation	Lipide	Protide	Glucide
1	Imc20-25	Imc20-25	Imc30+	Non	Manutention	Non	Équilibrée	Raisonnable	Raisonnable	Raisonnable
2	Imc30+	Imc30+	Imc30+	Oui	Cadre	Oui	Équilibrée	Raisonnable	Raisonnable	Raisonnable
3	Imc20-25	Imc20-25	Imc20-25	Oui	Cadre	Oui	Équilibrée	Raisonnable	Raisonnable	Raisonnable
4	Imc30+	Imc30+	Imc25-30	Non	Cadre	Non	Non équilibrée	Excès	Insuffisant	Excès
5	Imc20-25	Imc30+	Imc20-25	Non	Manutention	Oui	Équilibrée	Raisonnable	Raisonnable	Raisonnable
6	Imc30+	Imc25-30	Imc30+	Oui	Cadre	Non	Non équilibrée	Excès	Excès	Raisonnable

Apprentissage de la structure du réseaux Bayésien

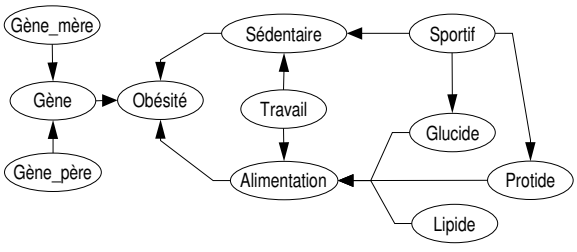


Figure 6.2.1 - Construction du réseau Bayésien

Les attributs caractérisant les domaines de la génétique, la nutrition et la sociologie sont respectivement Att_G = {gène, gène_mère, gène_père}, Att_N = {alimentation, lipide, glucide, protide} et Att_S = {sédentaire, travail, sportif}. Les bases de cas des trois experts sont représentées dans la Figure 6.2.1, elles contiennent les cas relatifs de l'expert décrit par les

attributs du domaine auquel il appartient. Les valeurs associées aux attributs appartiennent à l'ensemble de définition de cet attribut.

La fusion de ces bases de cas permet de regrouper les données relatives à un même patient. Cette nouvelle base de cas décrit les patients selon les attributs de $Att_G \cup Att_S \cup Att_N$, elle sert d'ensemble d'apprentissage et permet d'apprendre la structure du réseau de la Figure 6.2.1. Ce réseau autorise à diagnostiquer l'état du patient à l'aide de faits appartenant aux différents domaines. En effet, on peut déterminer à la fois les effets et les causes entraînant l'obésité du patient à l'aide des tables de probabilités conditionnelles et des probabilités des attributs de $Att_G \cup Att_S \cup Att_N$.

6.2.2 Processus de diagnostic

Le réseau Bayésien construit précédemment nous permet en fonction d'un ensemble de faits $F \subseteq 2^{Att_{D_i}(L) \times U_{D_i}}$, d'associer une probabilité à un attribut cible. Cet attribut cible représente soit la typologie que l'on souhaite attribuer à l'individu, soit un attribut quelconque de l'ensemble $Att_{D_i}(L)$ pour lequel on désire connaître la valeur la plus probable. Les faits, décrivant en partie ou en totalité le cas observé, modifient les probabilités des attributs par la propagation des probabilités dans le réseau. Néanmoins, certaines causes impactant sur l'attribut cible restent inconnues. Ces causes sont dites d-connectées à l'attribut cible, i.e. que la probabilité de l'attribut cible est conditionnellement dépendante des probabilités de ces causes (cf. [21]).

Algorithme 6.2.I : *Diagnostic Global*

ENTRÉE :

- F : l'ensemble des faits caractérisant l'individu.
- *net* : le réseau Bayésien.
  - *d-connexion(a)* récupère les attributs d-connecté à l'attribut a dans *net*;
  - *estObserve(a,v)* permet d'attribuer la probabilité *P(a=v)=1* à *net*;
  - *propagationProbabilités()* propage les probabilités dans net;
  - *probabiliteMaximale(a)* récupère la valeur *v* pour laquelle *P(a=v)* est maximale dans *net*.
- *target* : l'attribut cible.

SORTIE :

- L'argumentaire en faveur de la valeur la plus probable de *target*.

DiagnosticGlobal(F, net, target) :

1. Pour tout $(a, v) \in \mathsf{F}$ faire
2. *net.estObserve(a,v)*
3. Fin Pour
4. $Causes \leftarrow net.d-connexion(target)$
5. Pour tout $a \in Causes$ faire
6. $v \leftarrow valeur\ acceptable\ de\ a\ selon\ \Delta^{\mathsf{D}_i}(\mathsf{F}_{\mathsf{D}_i}, a)\ tel\ que\ \mathsf{Att} \cap \mathsf{Att}_{\mathsf{D}_i} \neq \emptyset, \forall i \in \{i, ..., n\}$
7. *net.estObserve(a,v)*
8. Fin Pour
9. *net.propagationProbabilités()*
10. $v_{target} \leftarrow net.probabilitéMaximale(target)$
11. Retouner $\langle (target, v_{target}), \bigcup_{i=1}^{n} \Delta^{\mathsf{D}_i}(\mathsf{F}_{\mathsf{D}_i}, a_i), net \rangle$

Le diagnostic global consiste à inférer les valeurs des attributs a_i d-connectés à l'attribut cible pour déterminer la probabilité de ce dernier (cf. Algorithme 6.2.I).

- Pour cela, nous faisons appel aux connaissances des agents du domaine D_i tel que $a_i \in \mathsf{Att} \cap \mathsf{Att}_{\mathsf{D}_i}$. Ainsi, pour chaque attribut a_i, un diagnostic local $\Delta^{\mathsf{D}_i}(\mathsf{F}_{\mathsf{D}_i}, a_i)$ est demandé auprès de l'agent Superviseur du domaine D_i où $\mathsf{F}_{\mathsf{D}_i} \subseteq \mathsf{F}$ est l'ensemble des faits concernant D_i (cf. *ligne 6).*
- La valeur v_i retournée par l'agent Superviseur est la valeur jugée acceptable par les agents du domaine D_i. Puisqu'elle a été déduite de l'ensemble des faits $\mathsf{F}_{\mathsf{D}_i}$, nous la considérons également comme la plus probable. Après que les diagnostics locaux aient été établis, les probabilités $P(a_i=v_i)$ sont considérées comme maximales, i.e. $P(a_i=v_i)=1$ (cf. *ligne 7*).
- Dans le cas où le diagnostic local n'a pas abouti, la probabilité de a_i reste inchangée. Les probabilités sont ensuite propagées dans le réseaux Bayésien et la probabilité du thème calculée (cf. *ligne 9* et *10*).
- La valeur du thème est alors justifiée par les dépendances causales décrites dans le réseau mais également par les diagnostics $\Delta^{\mathsf{D}_i}(\mathsf{F}_{\mathsf{D}_i}, a_i)$ construits localement par les agents du domaines. Ensemble, ces informations permettent d'expliquer le raisonnement effectué au niveau du diagnostic global.

Le processus de diagnostic global, décrit dans l'Algorithme 6.2.I, est réalisé par l'agent Bayésien. Cet agent, dont le comportement est détaillé dans la section 6.3.1, gère les échanges

avec les Agents Superviseur auprès desquels il récupère les diagnostics $\Delta^{D_i}(F_{D_i}, a_i)$ qui permettent de calculer les probabilités $P(a_i=v_i)$.

Exemple 6.2.II [Diagnostic global]

> Nous élaborons le diagnostic global d'un patient en utilisant le réseau Bayésien de la Figure 6.2.1. Plus exactement, nous cherchons à connaître la sévérité de son obésité. L'attribut cible est donc le noeud *Obésité*. Nous détenons néanmoins certaines informations sur le patient, nous savons qu'il est sédentaire, qu'il mange sainement et qu'un membre de sa famille est atteint d'obésité. L'ensemble des faits est alors $F=\{(Sédentaire, oui), (Alimentation, équilibrée), (Gène\_père, imc30+)\}$.
>
> La propriété de d-connexion nous permet de savoir que les attributs *Gène*, *Gène_mère* impactent la probabilité de l'attribut cible *Obésité*. L'Agent Superviseur appartenant au domaine de la génétique est impliqué dans le processus. Et, il construit à l'aide des Agents du domaine en génétique les diagnostics locaux $\Delta^{G}(F_G, Gène)$, $\Delta^{G}(F_G, Gène\_mère)$ où $F_G=\{(Gène\_père, imc30+)\}$.
>
> D'après sa base de cas (cf. Exemple 6.2.I), le généticien a pu construire une règle *r* telle que $r: Gène=imc30+ \leftarrow Gène\_père=imc30+$. L'Agent du domaine, représentant l'unique généticien du domaine, a pu construire un argument acceptable en faveur de *Gène* tel que $P=\langle Gène=imc30+, \{r, (Gène\_père=imc30+)\}\rangle$. Par conséquent, la probabilité $P(Gène=imc30+)=1$. Désormais, on connaît les probabilités des causes agissant directement sur le noeud *Obésité*, sa probabilité peut être calculée de façon plus précise.

6.3 Architecture

Les connaissances des agents concernent uniquement leur domaine d'expertise. Par conséquent, ils peuvent difficilement mettre en doute ou évaluer des arguments issus d'autres domaines. C'est pour cette raison que nous avons décidé de regrouper les agents du domaine par domaine.

Toutefois, le problème considéré est relatif à ces différents domaines. L'élaboration d'un diagnostic global (cf. section 6.2) repose alors sur les connaissances de chaque agent. À cet effet, nous utilisons le réseau Bayésien, construit à partir des bases de cas des agents, pour décrire les dépendances causales entre les attributs de différents domaines et proposer un diagnostic.

La gestion du réseau Bayésien est déléguée à un agent dit Bayésien. Cet agent a également pour tâche d'interagir avec les agents Superviseur de chacun des domaines afin

d'élaborer le diagnostic global. Cet agent et le protocole utilisé pour la construction du diagnostic global sont respectivement présentés dans les sections 6.3.1 et 6.3.2. Enfin, dans la section 6.3.3, l'architecture détaillée du système est introduite.

6.3.1 Agent Bayésien

Les réseaux Bayésiens permettent de représenter les connaissances de façon intuitive. Ces derniers permettent également de construire des arguments [20][76][118] et de leur associer une probabilité calculée à la fois avec les faits et les tables de probabilités. Par conséquent, ils peuvent servir à modéliser un diagnostic et expliquer son déroulement notamment à l'aide des dépendances causales.

De telles qualités font des réseaux Bayésiens un outil de diagnostic offrant à l'utilisateur la possibilité d'interpréter et d'évaluer les explications ainsi fournies (cf. Chapitre 1). C'est pourquoi l'Agent Bayésien, dont le comportement est schématisé dans la Figure 6.3.1, est également utilisé comme interface utilisateur. Les faits permettant les calculs de probabilités sont proposés par l'utilisateur. Chaque fait *(a, v)* est ensuite communiqué par l'Agent Superviseur du domaine $\mathrm{Att_D}$ tel que $a \in \mathrm{Att_D}$ (*Transmettre les faits*). Ces faits seront propagés au niveau des Agents du domaine et serviront à construire le diagnostic local.

Le diagnostic global (cf. section 6.2) nécessite l'avis des Agents du domaine sur les attributs d-connectés à l'attribut cible. Ces attributs sont le résultat d'un calcul basé sur la propriété de d-connexion et les faits précédemment introduits dans le réseau Bayésien (*Rechercher les sous-thèmes*). Pour chacun de ces attributs $a_i \in D$, un diagnostic local $\Delta^{D}(F_D, a_i)$ complète les inférences réalisées dans le réseau Bayésien. Ces diagnostics locaux sont demandés par l'Agent Bayésien (*Transmettre les sous-thèmes*). Dans le cas ou le diagnostic $\Delta^{D}(F_D, a_i)$ n'a pas pu être établi (Certains avis manquants), de nouvelles demandes concernant les attributs d-connectés à a_i seront envoyés par l'Agent Bayésien.

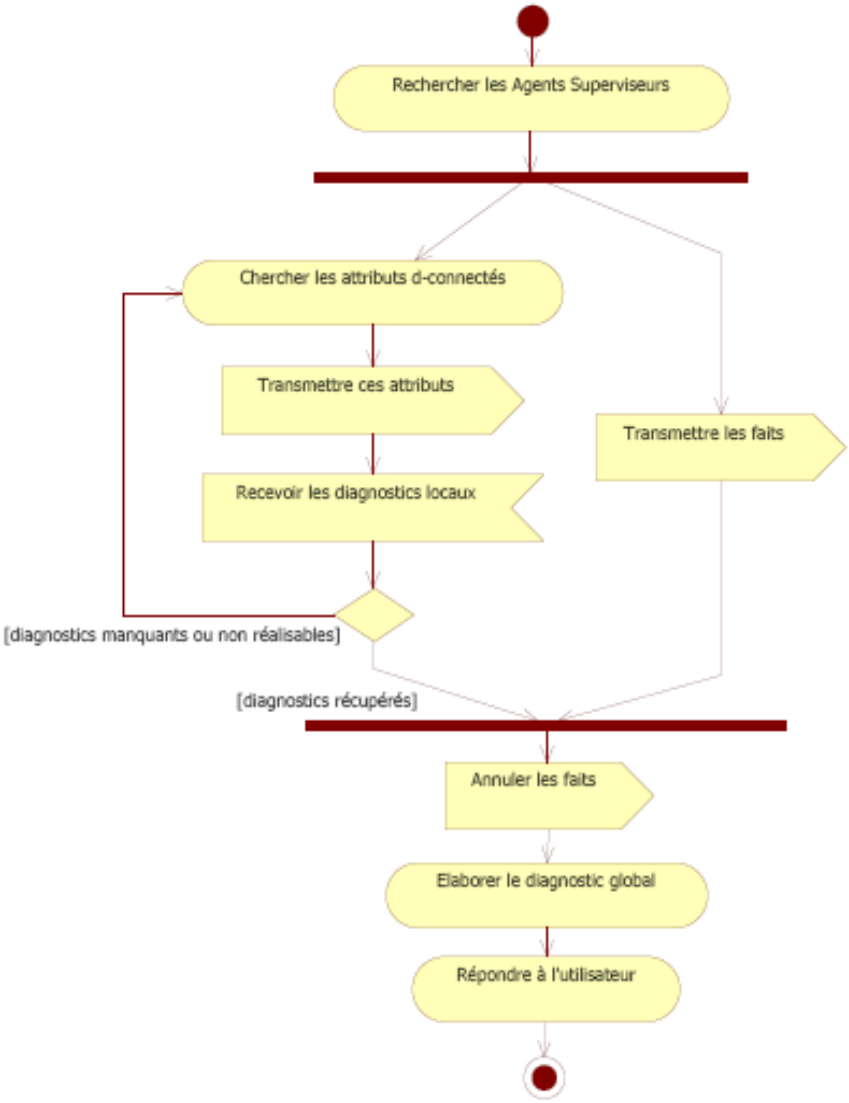


Figure 6.3.1 - Comportement de l'Agent Bayésien

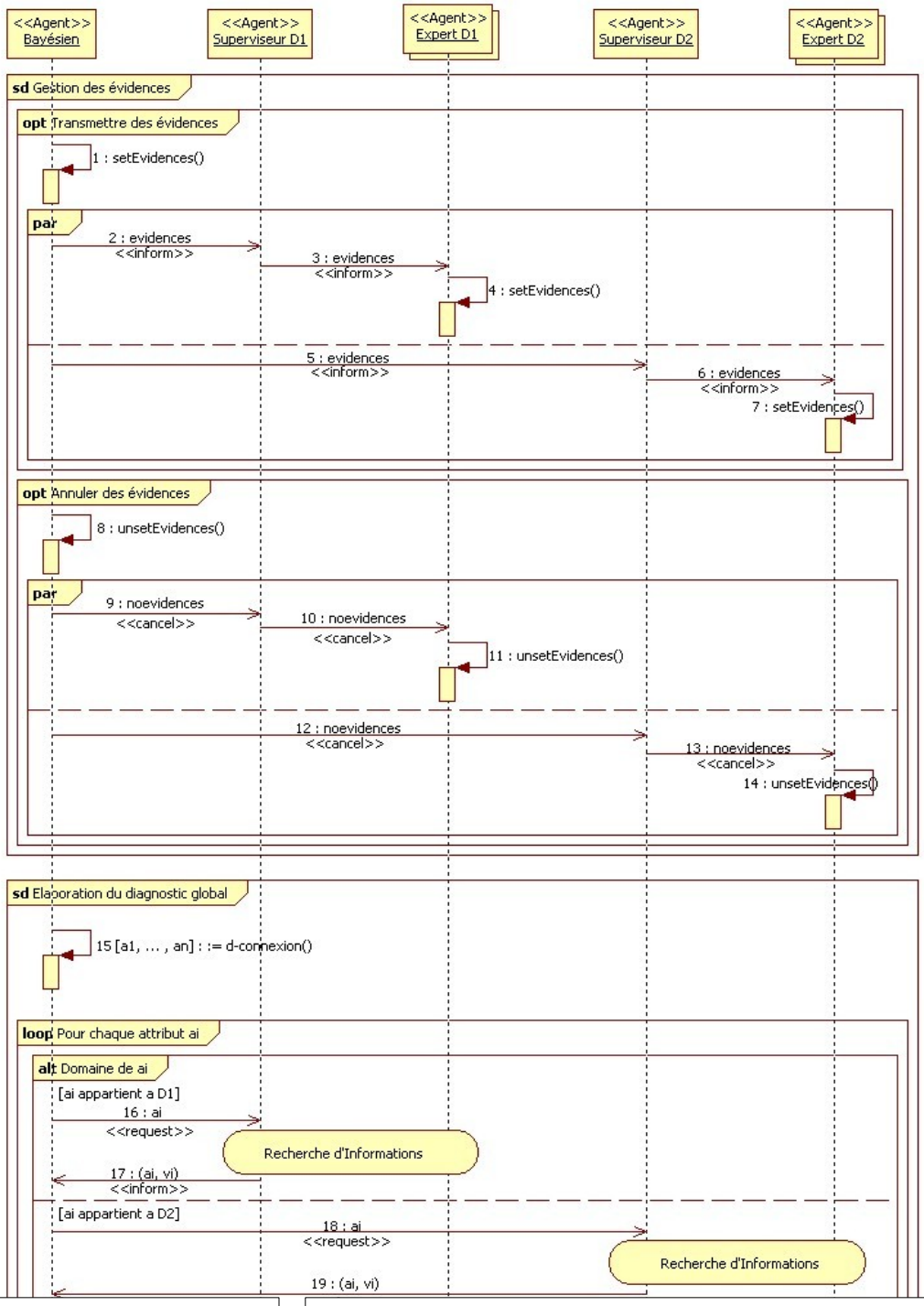
Ces diagnostics déterminent la valeur la plus probable des attributs a_i. Par conséquent, l'attribution de telles valeurs peut impacter sur la valeur associée de l'attribut cible. C'est pour cette raison que l'Agent Bayésien attend la réception des diagnostics locaux (*Après tous les avis reçus*) avant de *Propager les probabilités dans le réseau*. Le réseau Bayésien ainsi instancié et les diagnostics locaux fournissent à l'utilisateur les explications nécessaires pour comprendre le raisonnement effectué par les agents.

Le protocole de diagnostic autorise la réalisation du diagnostic global et formalise les interactions décrites précédemment. Ce protocole schématisé dans la Figure 6.3.2 est décrit dans la section suivante.

6.3.2 Protocole de diagnostic

Le protocole de diagnostic (cf. Figure 6.3.2) formalise les interactions entre l'agent Bayésien et les agents Superviseurs avec lesquels il communique directement. Pour plus de clarté, le protocole présenté ici décrit uniquement les interactions entre l'agent Bayésien et les agents (un agent Superviseur et un agent du domaine) de deux domaines distincts.

Les deux tâches nécessitant une interaction sont respectivement la gestion des faits (regroupant *« Transmettre le faits »* et *« Annuler les faits »*) et l'élaboration du diagnostic global. Comme précisé ci-avant, les faits introduits par l'utilisateur sont propagés jusqu'aux agents du domaine du domaine concerné. Cette propagation, s'effectuant pour chaque domaine en parallèle, est décrite à l'aide de deux messages *inform* contenant les faits précédemment introduits (messages n°2/5 et n°3/6). De façon similaire, l'agent Bayésien peut alerter les agents de l'invalidité des faits. Cette fois-ci, la propagation est réalisée à l'aide d'un message *cancel* dont le contenu décrit les faits désormais invalides (message n°9/12 et n°10/13). L'utilisateur pouvant ajouter ou supprimer des faits de façon indépendante ces interactions sont optionnelles.




```
type = BNA
gui = true
net = kb/bayesnet.xml
dom = kb/nodestype.xml
```

```
type = Expert
domain = Nutrition
gui = false
behaviour = agent.behaviours.ExpertBehaviour
kbase = kb/kbAE1.xml
model = gad.calcul.DefaultProbaModel
```

Figure 6.3.3 - Fichier de properties de l'Agent Bayésien et d'un Agent Expert

Figure 6.3.2 - Protocole pour le diagnostic global

L'élaboration du diagnostic global débute avec le calcul des attributs d-connectés à l'attribut cible (*« Chercher les attributs d-connectés »*). Les attributs ainsi calculés font l'objet d'un diagnostic local. Selon le domaine de l'attribut, l'Agent Bayésien contacte l'Agent Superviseur D1 ou D2. Sa demande se présente sous la forme d'un message *request* (message n°16 et n°18) dont le contenu est l'attribut en question. Le diagnostic local est caractérisé par l'appel au protocole de Recherche d'informations. Une fois les diagnostics locaux établis (*« Recevoir les diagnostics locaux »*), l'Agent Bayésien propage les probabilités dans le réseau (stimulus n°20) et récupère la probabilité $P(a=v)$ telle que a est l'attribut cible et v la valeur la plus probable que l'attribut a peut prendre (stimulus n°21).

6.3.3 Architecture détaillée

Le modèle statistique représente le coeur du système. En effet, il met en relation les différents domaines et offre dès lors un moyen de diagnostiquer un cas (individu ou système) en considérant ces relations. L'Agent Bayésien a été introduit afin de superviser les interactions. Ce dernier gère, en fonction des faits relatifs au cas, les demandes de diagnostics locaux nécessaires à l'élaboration du diagnostic global.

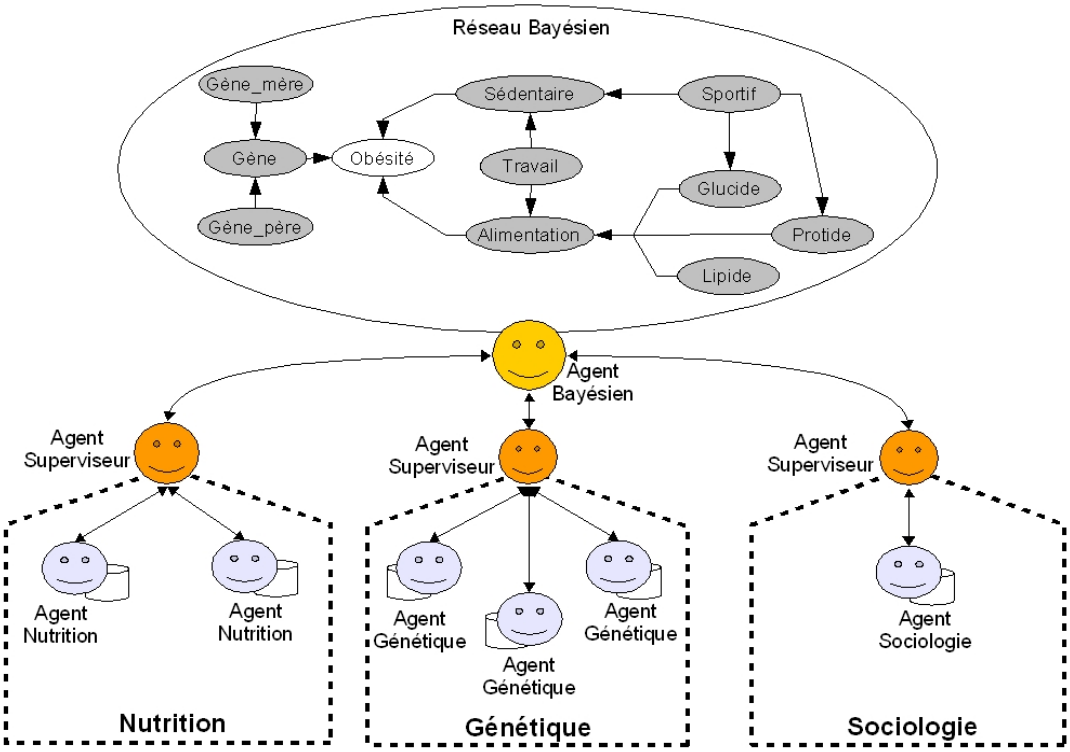


Figure 6.3.4 - Architecture détaillée du système

Dans ce but, il interagit avec les Agents Superviseur qui dirigent la construction de l'argumentaire dans leurs domaines respectifs (cf. Chapitre 5). Chaque domaine D est composé d'un Agent Superviseur et d'un ensemble d'Agents du domaine D dont les connaissances concernent uniquement le domaine D. L'Agent Superviseur représente le point d'accès au domaine alors que les Agents du domaine représentent les experts humains au sein du système.

La Figure 6.3.4 schématise l'architecture présentée dans l'Exemple 6.2.I. L'attribut cible du réseau Bayésien est le noeud *Obésité*. L'obésité comme expliqué précédemment est issue de plusieurs domaines tels que la *Nutrition*, la *Génétique* et la *Sociologie*. De ce fait, il existe trois Agents Superviseur gérant chacun un de ces domaines. Les interactions entre les agents sont représentées par les flèches bilatérales. Les éléments accolés aux Agent du domaines symbolisent leur base de connaissances.

6.4 Synthèse

L'introduction d'un modèle statistique permet de mettre en correspondance les attributs de divers domaines. On peut dès lors mettre en exergue les effets d'une décision prise par les experts d'un domaine sur un attribut d'un autre domaine. Un diagnostic, dit global est alors établi en se servant des caractéristiques du cas considéré. Ces caractéristiques sont considérés comme des faits et nous donnent la possibilité d'attribuer une valeur à l'attribut cible en fonction de la probabilité calculée par propagation.

Néanmoins, les faits introduits dans le réseau Bayésien ne correspondent pas toujours à l'ensemble des causes impliquant directement l'attribut cible. Nous utilisons alors les connaissances des experts, représentés par les Agents du domaine, pour déduire certaines des valeurs manquantes. Ces valeurs sont alors justifiées par un argumentaire résultant d'un échange d'arguments entre les Agents du domaine d'un même domaine. Les éléments sur lesquels se basent les argumentaires des Agents du domaine sont également les faits introduits initialement dans le réseau. En agissant ainsi, nous attribuons à l'attribut cible une valeur justifiée à la fois par les faits, la structure du réseau et les connaissances de chaque Agent du domaine présent dans le système multi-agent. Le diagnostic global correspond alors à l'ensemble des explications justifiant la valeur de l'attribut cible.

Conclusion du modèle

Dans cette partie, nous avons proposé un outil de diagnostic pluridisciplinaire dans lequel des agents coopèrent ensemble pour élaborer des diagnostics propres à leur domaine d'expertise (ou diagnostics locaux) avant de proposer un diagnostic global du système.

Nous avons proposé, dans le Chapitre 4, une logique argumentative qui constitue le modèle de raisonnement des agents. Cette logique argumentative permet de gérer les interactions entre des arguments contradictoires mais également, d'adapter le raisonnement de l'agent en fonction des évidences présentes dans le système. Pour cela, elle comprend une relation de défaite basée à la fois sur une relation d'attaque et une relation d'ordre. La relation d'attaque s'intéresse à déterminer si deux arguments sont contradictoires. Dans le cas ou les arguments sont contradictoires, nous utilisons une la relation d'ordre afin de les départager. La relation d'ordre que nous proposons permet d'attribuer une force aux arguments en fonction i) de la base de cas de l'expert, ii) de la base de connaissances de l'agent qui représente l'expert en question et iii) des évidences présentes dans le système ou son environnement. Ainsi, nous sommes en mesure de construire et d'attribuer dynamiquement une force aux arguments en fonction des évidences.

Dans le Chapitre 5, nous avons détaillé comment nous élaborions un diagnostic local. Ce diagnostic est le résultat d'un processus d'argumentation durant lequel les agents d'un même domaine d'expertise s'échangent des arguments construits et évalués grâce à la logique argumentative introduite dans le Chapitre 4. Dans le but d'obtenir un argumentaire complet, nous avons mis à la disposition des agents une mémoire partagée intelligente : l'Agent Superviseur (cf. Chapitre 5). Cet agent dirige le débat, fournit les informations souhaitées aux agents et centralise leurs avis. Ainsi, il nous permet de préserver l'expertise des experts humains (apprise à partir des données ou définie à l'aide de l'expert humain) qui est retranscrite au sein des bases de connaissances des agents.

Enfin, nous avons présenté dans le Chapitre 6, l'élaboration du diagnostic global. Ce diagnostic est basé sur un modèle statistique et les diagnostics locaux réalisés précédemment

par les Agents de domaines (cf. Chapitre 5). L'intégration des diagnostics locaux dans le réseau Bayésien permet d'instancier les probabilités des noeuds dont la valeur est inconnue. La probabilité du noeud qui représente le défaut est alors justifiée par les diagnostics locaux et les relations de cause à effet présentes dans le réseau Bayésien.

Le modèle ANDi proposé ici est la base de deux outils de diagnostic servant respectivement pour le suivi de personnes souffrant de surpoids et le diagnostic de défauts dans un laminoir circulaire. Ces outils de diagnostic sont présentés dans la partie suivante.

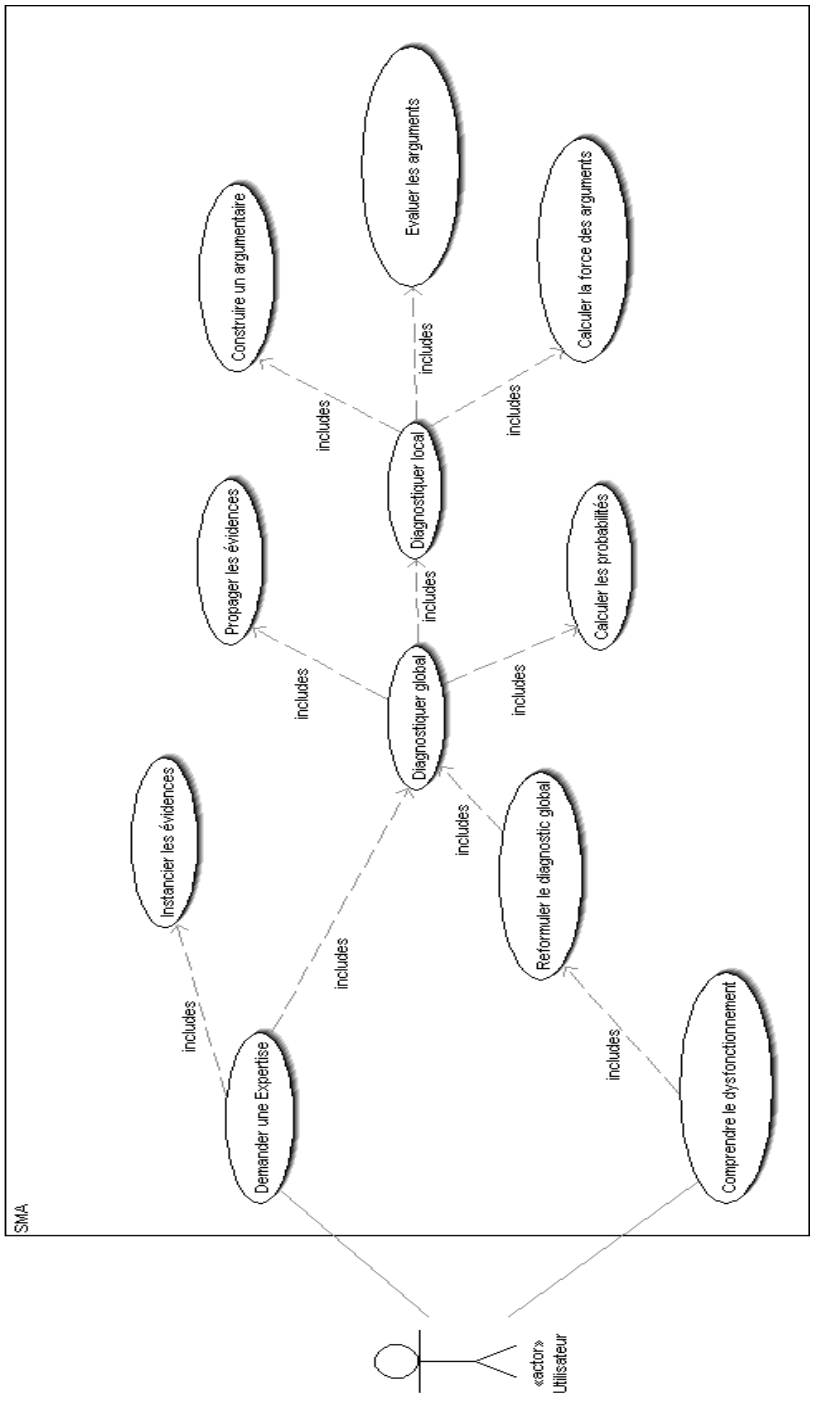


Figure 1 - Diagramme de cas du SMA

Troisième partie

Mise en oeuvre et réalisation

Chapitre 7

Mise en oeuvre et réalisations

Sommaire

7.1 Introduction

Dans ce chapitre, nous présentons l'outil de diagnostic réalisé à l'aide du modèle ANDi décrit dans la partie précédente. L'objectif d'un tel système est de modéliser les connaissances d'une communauté d'experts pour réaliser le diagnostic d'un problème pluridisciplinaire.

Ce système de diagnostic a été appliqué à deux cas d'application : la télé-consultation de l'obésité et le laminoir circulaire. Nous proposons, dans chacun de ces cas, un outil permettant d'expliquer à l'utilisateur le ou les problèmes existants. Plus précisément, nous cherchons, dans le cadre de la télé-consultation, à responsabiliser l'utilisateur afin d'éviter des écarts au niveau de sa thérapie. L'application du laminoir circulaire, quand à elle, tente d'apporter et d'expliquer les défauts survenus lors du laminage d'une pièce. En effet, la correction des défauts lors du laminage se base sur l'intuition et l'expérience du lamineur, et ne peut-être ni anticipée ni évitée.

Dans un premier temps, nous allons décrire la mise en oeuvre de l'outil de diagnostic (cf. section 7.2). Puis, nous présenterons les deux contextes d'application auxquels nous nous sommes intéressés (cf. section 7.3).

7.2 Mise en oeuvre de ANDi

Nous allons fournir dans cette section des éléments permettant la mise en oeuvre de ANDi. Dans un premier temps, nous allons apporter de plus amples informations sur l'environnement et les algorithmes utilisés pour l'extraction de connaissances (cf. section 7.2.1). Puis, nous décrirons le système multi-agents qui a été mis en oeuvre (cf. section 7.2.2) pour répondre au besoin de ces deux cas d'application.

7.2.1 Extraction des connaissances

L'extraction de connaissances et le pré-traitement des données ont été réalisées avec WEKA (Waikato Environment for Knowledge Analysis). WEKA est un environnement, entièrement développé en Java, conçu pour l'analyse de données et l'extraction de connaissances. Il permet de pré-traiter les données, de les analyser à l'aide d'une méthode d'apprentissage (à choisir parmi une vingtaine d'algorithmes) et, d'afficher à la fois le modèle appris et ses performances. Ce logiciel réalisé par l'Université du Waikato (Nouvelle-Zélande) est actuellement distribué sous licence publique GNU. De par sa conception, WEKA autorise l'intégration de nouveaux algorithmes d'apprentissage. Les contributions extérieures et l'implication de l'équipe du Waikato expliquent notamment l'impressionnante collection d'algorithmes déjà développés et intégrés dans WEKA. Par ailleurs, WEKA comprend un environnement d'analyse inspiré des flux de données (Data-Flow). Cet environnement, appelé « *Knowledge Flow Environment*», a pour but la création de processus d'apprentissage sous forme graphique. De tels processus sont utiles pour l'automatisation et la réutilisation du procédé d'apprentissage. Ces processus, au même titre que les algorithmes, peuvent être appliqués directement à un ensemble de données par le biais des interfaces graphiques de WEKA ou intégrés dans du code JAVA.

Dans le cadre de la thèse, nous nous intéressons à WEKA plus particulièrement pour la découverte de typologies (uniquement dans le cadre de la télé-consultation de l'obésité), la découverte de règles d'associations et l'apprentissage des réseaux Bayésiens. À cet effet, nous utilisons l'interface définie dans WEKA pour fouiller les données. L'automatisation du procédé n'étant pas notre priorité, nous avons pour l'instant mis de côté la définition des processus par le biais du « *Knowledge Flow Environment*».

Dans un premier temps, nous avons pré-traité les données pour les rendre exploitables par WEKA. Le pré-traitement des données consiste à une discrétisation non supervisée des attributs

numériques contenus dans l'ensemble de données. Les valeurs de ces attributs ont été regroupées en 25 intervalles (ou *bins*) décrits par des étiquettes. Ce pré-traitement est nécessaire à l'utilisation des algorithmes *APriori* et *BayesNet* qui requièrent des valeurs nominales. Pour discrétiser nos données, nous avons utilisé un algorithme de filtrage non-supervisé, appelé *Discretize*.

L'algorithme *X-Means* [85] est une extension de l'algorithme de clustering *K-Means*. Cet algorithme a été appliqué à nos données (non discrétisées) dans le but de découvrir les typologies de l'obésité. Certaines typologies étaient a-priori connues des experts médicaux, l'utilisation du clustering a donc permis de vérifier leur pertinence. Par ailleurs, ces typologies ont permis de paramétrer le nombre minimal de clusters de l'algorithme *X-Means*. Le nombre maximal a été évalué suite à des tests.

Nous avons ensuite utilisé l'algorithme *APriori* pour la découverte des règles d'associations. Cet algorithme, proposé par Agrawal et Srikant [3], permet de découvrir des relations pertinentes entre les attributs d'une large base de données. Les règles ainsi construites sont évaluées à l'aide d'un seuil de confiance fixé à *0,8*. De nouvelles règles possédant la même structure peuvent être proposées par un expert humain. Ensembles, les règles d'associations et les règles issues des experts composent la base de connaissances des Agents de domaine.

Pour finir, le réseau Bayésien est le résultat de l'algorithme d'apprentissage *BayesNet*. Cet algorithme prend comme paramètres deux algorithmes : i) un estimateur utilisé pour générer les tables de probabilités conditionnelles et ii) un algorithme de recherche utilisé pour apprendre la structure du réseau. L'ensemble d'apprentissage étant conséquent, nous avons opté pour des algorithmes relativement peu gourmands en mémoire : *SimpleEstimator* et *TabuSearch* [17]. L'algorithme *SimpleEstimator* instancie les tables de probabilités conditionnelles directement à partir des données. Cette instanciation est réalisée après que la structure du réseau Bayésien ait été apprise par l'algorithme *TabuSearch*. L'algorithme *TabuSearch* est basé sur la technique d'optimisation du *Hill Climbing* [103] et consiste à rechercher une solution optimale par modifications successives (ajout, retrait ou inversion des arcs) d'une solution quelconque. Après validation, le réseau Bayésien ainsi appris est mis à la disposition de l'Agent Bayésien.

7.2.2 Système multi-agents

Dans cette section, nous présentons l'architecture commune aux deux cas d'applications décrits précédemment. Nous commençons par expliquer le fonctionnement général du système multi-agents (cf. section 7.2.2.1). Ensuite, nous présentons brièvement la plateforme choisie afin de déployer les agents (cf. section 7.2.2.2) et nous expliquons les différents paramètres de configuration utilisés (cf. section 7.2.2.3). Enfin, nous introduisons succinctement le format de fichier utilisé pour décrire les bases de connaissances des agents (cf. section 7.2.2.4).

7.2.2.1 Fonctionnement général

Le fonctionnement général du système multi-agents est représenté par le cas d'utilisation de la Figure 1.2.4. Le système est destiné à aider l'utilisateur à comprendre le dysfonctionnement (*Comprendre le dysfonctionnement*). Pour cela, il réclame une expertise (*Demander une expertise*). Cette expertise repose sur les faits fournies par l'utilisateur (*Instancier les évidences*) et le diagnostic global (*Diagnostiquer global*) réalisé par l'Agent Bayésien. Comme expliqué dans le Chapitre 6, ce diagnostic global est la synthèse de plusieurs diagnostics locaux (*Diagnostiquer local*) permettant la propagation et le calcul des probabilités dans le réseau Bayésien (*Calculer les probabilités*). Les diagnostics locaux sont le résultat d'un argumentaire (*Construire un argumentaire*) réalisés par des Agents Expert et des Agents Superviseur. Un diagnostic local constitue à l'évaluation des arguments (*Evaluer les arguments*) proposés par les Agents Expert à l'aide de leur force (*Calculer la force des arguments*). Ces arguments sont également basés sur les évidences qui ont été propagées par l'Agent Bayésien puis les Agents Superviseur (*Propager les évidences*). L'explication du dysfonctionnement fournie à l'utilisateur consiste à reformuler les diagnostics réalisés par les agents (*Reformuler le diagnostic global*).

Les Agents Superviseur et les Agents Expert sont constitués de deux comportements. Le premier comportement permet de prendre en compte les évidences fournies par l'utilisateur. Le second comportement est dédié à la création et/ou l'évaluation des arguments. Ces comportements s'exécutent en parallèles. Les comportements associés à un Agent Superviseur sont respectivement appelés *SupervisorSubBehaviourEvidence* et *SupervisorSubBehaviour Acceptable*. Ceux associés à un Agent du domaine sont appelés *ExpertSubBehaviourEvidence* et *ExpertSubBehaviourAcceptable*. Les diagrammes de classe de l'Agent Superviseur et de l'Agent du domaine sont présentés dans les Figure 7.4.1 et Figure 7.4.2.

L'Agent Bayésien est lui composé d'un unique comportement, appelé *BayesNetBehaviour* (cf. Figure 7.4.4). Ce comportement lui permet de propager les probabilités dans le réseau Bayésien à la réception des diagnostics locaux. L'utilisateur a la possibilité d'instancier les fait par le biais de l'interface graphique de l'Agent Bayésien. Le résultat de la reformulation du diagnostic global est également affiché par le biais de cette interface. Les diagnostics locaux peuvent quant à eux être visualisés grâce aux interfaces graphiques des Agents Superviseur et des Agents de domaine. Ces interfaces permettent de visualiser le graphe argumentatif de l'agent considéré (cf. Figure 7.4.8) mais également, les arguments construits lors de l'argumentation et les valeurs acceptables des différents attributs (cf. Figure 7.4.9)

7.2.2.2 Plateforme Agent

Le système multi-agents a été développé à l'aide du framework JADE (Java Agent DEvelopment Framework). JADE permet de développer des applications basées sur la technologie « Agent » qui respectent les spécifications de la FIPA [REF ou url]. JADE est un logiciel distribué par Telecom Italia sous licence LGPL. Il est entièrement implémenté en Java et requiert au minimum la version 1.4 du JDK. C'est la raison pour laquelle nos agents sont implémentés en Java. Par ailleurs, JADE comprend une « plateforme Agent » pouvant être distribuée entre plusieurs machines. Les agents déployés sur cette plateforme communiquent ensemble à l'aide de messages, conformes à la syntaxe définie par la FIPA. Ainsi, les protocoles introduits dans les Chapitres 5 et 6 ont aisément pu être implémentés. De plus, JADE intègre un ensemble de services facilitant le développement. Parmi ces services, deux annuaires (pages blanches et pages jaunes) sont présents. Ils référencent respectivement les agents présents dans le système et les services qu'ils proposent.

7.2.2.3 Configuration des agents

La configuration des agents se fait par le biais d'un fichier *properties* (cf. Figure 7.3.4) qui spécifie les principales caractéristiques des agents. Le nombre de propriétés s'élève à huit :

1. ***type*** : le type spécifie la nature de l'agent. Trois types ont été définies : *BNA*, *Proxy* et *Expert.* Ces types sont respectivement associés aux Agents Bayésien, Superviseur et du domaine introduits dans les Chapitre 5 et 6.
2. ***domain*** *:* cette propriété spécifie le domaine d'expertise de l'agent. Elle est uniquement nécessaire aux agents de type de *Proxy* et *Expert*.
3. ***gui*** : les valeurs *true* et *false* peuvent être associées à la propriété *gui*. Lorsque la valeur de la propriété *gui* est égale à *true*, une interface graphique est lancée au démarrage de l'agent. L'interface d'un Agent Bayésien permet de : rentrer les faits, visualiser le réseau Bayésien qu'il gère, démarrer le processus de diagnostic (cf. Figure 7.4.6) et visualiser le résultat de la propagation des probabilités dans le réseau Bayésien avant et après l'argumentation (cf. Figure 7.4.7). Les interfaces des Agents Superviseur et des Agents du domaine (cf. Figure 7.4.8 et Figure 7.4.9) sont similaires. Elles permettent de visualiser i) la base de connaissances de l'agent, ii) les arguments qu'il juge acceptables et iii) son graphe argumentatif. Les informations affichées au travers de l'interface d'un Agent Superviseur sont celles acquises durant l'élaboration du diagnostic local. Ces interfaces ont été implémentées en Java et développées en *Swing*. L'affichage des réseaux Bayésiens et des graphes argumentatifs a été réalisée grâce aux API *JPowerGraph* et *PowerSing*.

kbase : cette propriété est uniquement disponible pour les Agents de domaine. Elle spécifie l'emplacement du fichier contenant la base de connaissances de l'agent. Cette base de connaissances est décrite à l'aide d'un fichier XML dont la syntaxe est expliquée dans la section 7.2.2.4. Le parser a été réalisé avec *SAX*. Nous représentons la base de connaissances à l'aide d'un graphe argumentatif. Le graphe argumentatif a été implémenté en Java, son diagramme de classe est représenté dans la Figure 7.4.4.

4. ***model*** : cette propriété est disponible uniquement pour les Agents de domaine. Elle spécifie le modèle de calcul à utiliser pour calculer la force des arguments. Nous avons implémenté le modèle de calcul présenté dans le Chapitre 4. Ce dernier se nomme *gad.calcul.DefaultProbaModel*. D'autres modèles de calcul peuvent être définis, ils doivent respecter la classe abstraite *gad.calcul.ProbabilityModel* (cf. diagramme de classes).

net : la propriété *net* spécifie l'emplacement du fichier contenant la représentation du réseau Bayésien. Le réseau Bayésien est décrit selon le format *BIF*[7] (BayesNet Interchange Format) proposé par les membres de l'UAI (Universal Agencies Inc.) pour augmenter et simplifier les échanges de résultats expérimentaux. Le format *BIF* est basé sur XML, nous utilisons l'API *JavaBayes* pour analyser les fichiers et manipuler les instances ainsi créées. Un exemple de fichier *BIF* est présenté dans la Figure 7.4.5.

dom : un type est associé aux noeuds du réseau Bayésien. Ce type correspond au domaine d'expertise auquel il appartient. L'Agent Bayésien se sert du type d'un noeud pour contacter l'Agent Superviseur du domaine et ainsi récupérer le diagnostic local qu'il a réalisé avec l'aide des Agents de domaine. Nous décrivons l'association des noeuds et de leur domaine à l'aide d'un fichier XML (cf. section 7.2.2.4). L'emplacement de ce fichier est la valeur de la propriété *dom*. Comme pour la base de connaissances, le parser a été réalisé avec *SAX*.

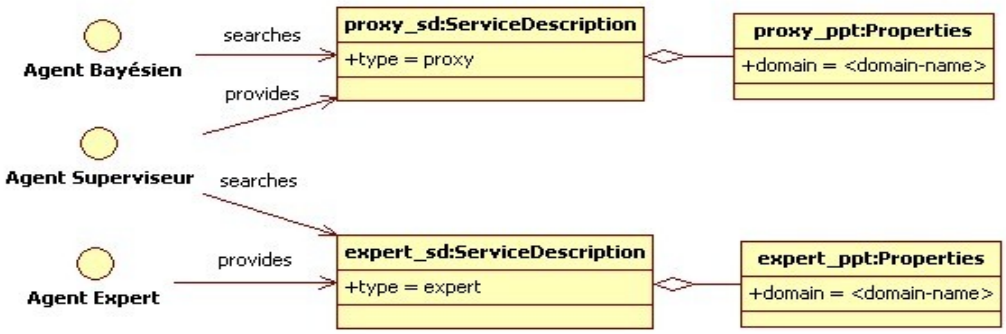


Figure 7.2.1: Description des services des agents

[7]http://www.cs.cmu.edu/afs/cs/user/fgcozman/www/Research/InterchangeFormat/

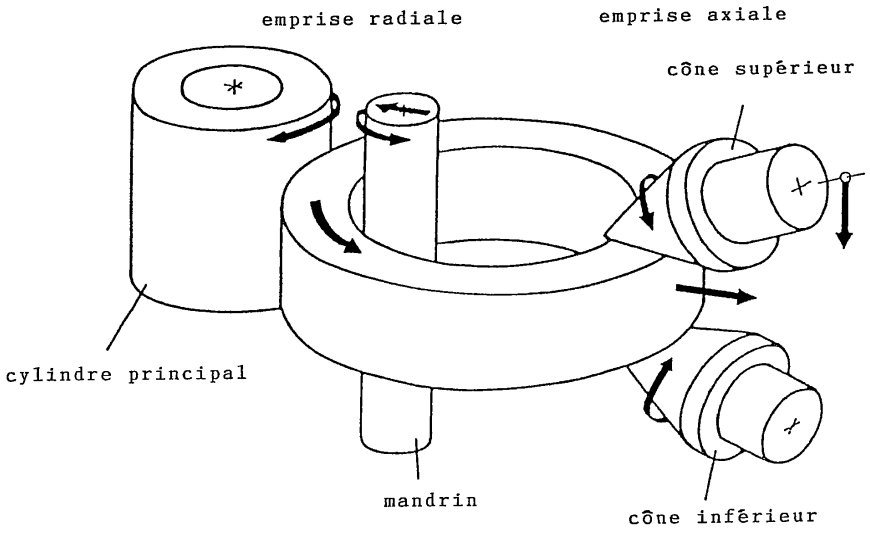


Figure 7.2.2 - Principe de laminage

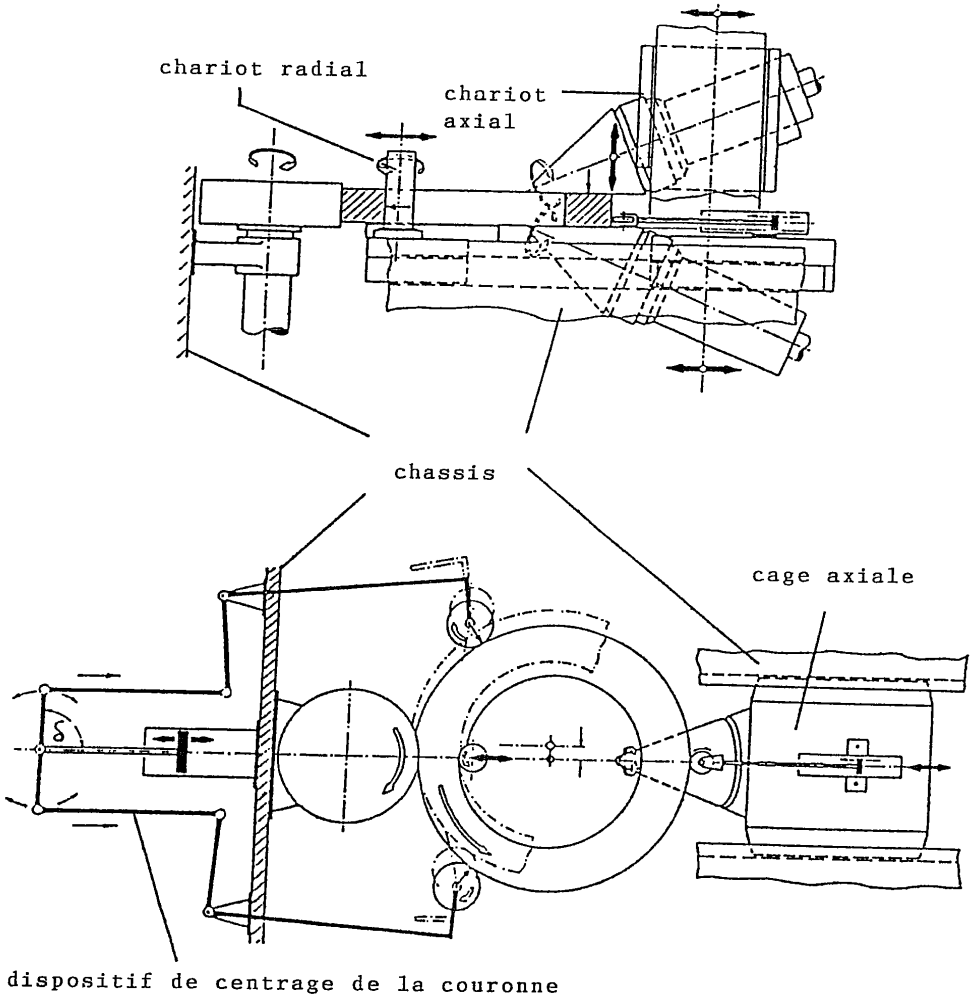


Figure 7.2.3 - Laminoir circulaire

5. ***behaviour*** : le fonctionnement des Agents Superviseur et des Agents de domaine est très proche. Alors, nous les avons regroupé dans une même classe appelée *ArgumentativeAgent* (cf. Figure 7.4.1). Cette classe instancie l'agent à l'aide du fichier *properties* et implémente les méthodes communes aux Agents Superviseur et aux Agents Expert. La propriété *behaviour* nous permet de spécifier le comportement implémentant le protocole de communication propre à l'agent considéré (cf. Chapitre 5 et 6). Les comportements des Agents Superviseur et des Agents Expert sont respectivement *agent.behaviours.SupervisorBehaviour* et *agent.behaviours.ExpertBehaviour*. Les comportements, introduits dans la section 7.2.2.1, sont les sous comportement des comportement *SupervisorBehaviour* et *ExpertBehaviour*. Cette propriété n'est pas présente dans le fichier *properties* de l'Agent Bayésien.

Les Agents de domaine et les Agents Superviseur s'enregistrent auprès du service de pages jaunes en utilisant leur type et leur domaine. De cette manière, il est aisé de retrouver les agents désirés. Les services référencés par les agents sont décrits dans la Figure 7.2.1.

7.2.2.4 Format de fichiers

La base de connaissances des Agents de domaine est enregistrée dans un fichier XML. La balise principale est la balise *KnowledgeBase*. À l'intérieur de cette balise sont imbriquées des balises *Rule* et *Frequencies*. Plusieurs balises *Rule* peuvent être définies, chacune de ses balises constitue une règle d'associations dont la conclusion et la prémisse sont respectivement définies à l'aide des balises *Conclusion* et *Premise*. Les éléments du langage sont représentés par la balise *Instance* dont les attributs *name* et *value* spécifient le nom et la valeur de l'élément. Au sein de la balise *Frequencies*, nous attribuons les probabilité *p(nom=valeur)* au couple $(nom, valeur) \in Att(L) \times U_{nom}$ ou nom est un élément du langage (cf. Chapitre 4).

Exemple 7.2.1 [Représentation d'une base de connaissances]

La règle d'association $r: (surpoids, true) \leftarrow (taille, petit) \wedge (poids, lourd)$ est représenté par le code XML ci-dessous. Les probabilités attribuées aux couples (*taille*, *petit*), (*poids*, *lourd*) et (*surpoids*, *true*) sont respectivement égales à *0.6*, *0.7* et *0.6*.

```
<KnowledgeBase>
  <Rule>
    <Conclusion>
        <Instance name="surpoids" value="true" />
    </Conclusion>
    <Premise>
        <Instance name="taille" value="petit" />
        <Instance name="poids" value="lourd" />
    </Premise>
  </Rule>
  <Frequencies>
    <Instance name="taille" value="petit" frequency=".6" />
    <Instance name="poids" value="lourd" frequency=".7" />
    <Instance name="surpoids" value="true" frequency=".6" />
  </Frequencies>
</KnowledgeBase>
```

Précédemment, nous avons énoncé, la présence d'un fichier XML permettant de typer les noeuds du réseau Bayésien. La balise principale de ce fichier est NodeDescription, elle encapsule un nombre de balises *Node* égal au nombre de noeuds du réseau Bayésien. Les attributs *name* et *domain* de la balise *Node* permettent d'associer le noeud portant le nom *name* au domaine *domain*.

Exemple 7.2.2 [Typage du réseau]

Dans le fichier ci-dessous, on associe les noeuds *cardiaque* et *santé* au domaine de la Nutrition. De façon similaire, nous associons respectivement les noeuds *sport* et *stress* aux domaines *Physiologie* et *Psychologie*.

```
<NodeDescription>
  <Node name="cardiaque" domain="Nutrition" />
  <Node name="sante" domain="Nutrition" />
  <Node name="sport" domain="Physiologie" />
  <Node name="stress" domain="Psychologie" />
</NodeDescription>
```

7.3 Expérimentations

Dans cette section, nous présentons les deux contextes auxquels nous nous intéressons. Nous commençons par décrire la télé-consultation de l'obésité (cf. section 7.3.1) puis le

laminoir circulaire (cf. section 7.3.2). La mise en oeuvre du système de télé-consultation de l'obésité nécessitant un investissement en temps trop important : la mise en place du site, le suivi d'une cohorte, l'analyse, l'extraction et la validation des données, etc; nous présenterons uniquement les résultats obtenus dans le cadre du laminoir circulaire.

7.3.1 Télé-consultation de l'obésité

Le sujet de la télé-consultation de l'obésité s'intègre à un projet du C.H.U. de Saint Étienne. Initialement, ce projet consiste à l'installation d'une télé-balance pour le suivi à domicile de patients atteints d'hypertension. Cette balance est destinée à suivre la courbe de poids des patients. En effet, une crise d'hypertension est caractérisée par une prise de poids rapide, pouvant aller de cinq à dix kilos par semaine. L'idée est d'utiliser l'infrastructure existante pour diversifier les services de suivi à domicile de la télé-balance. Cependant, le coût de production de la balance est élevé et son ergonomie restreinte. C'est une des raisons pour lesquelles l'application a été déployée sur un objet portable.

D'autres services intégrant la télé-balance sont envisageables. Nous focalisons notre attention sur l'obésité à cause du problème grandissant qu'elle représente (cf. section 7.3.1.1). L'obésité est une maladie mal connue des médecins dû aux nombreuses causes médicales auxquelles cette dernière est liée (nutrition, psychologie, etc) mobilisant de multiples spécialités. Notre démarche (cf. section 7.3.1.2) consiste principalement à proposer un outil permettant de croiser les spécialités et de mieux connaître l'obésité. Dans ce but, nous cherchons à acquérir des données issues d'un ensemble de patients (cf. section 7.3.1.3) pour extraire, à l'aide d'algorithmes d'apprentissage automatique, de nouvelles connaissances (cf. section 7.3.1.3.1). Ces nouvelles connaissances sont ensuite utilisées pour proposer un suivi personnalisé à chaque patient (cf. section 7.3.1.3.2). L'application permettant l'acquisition de données et le suivi des patients est appelée Compagnon Santé. L'architecture du Compagnon Santé et du réseau dans lequel il s'insère sont présentées dans la section 7.3.1.4.

7.3.1.1 Contexte et motivations

En 2000, les résultats de l'enquête nationale *ObEpi* [83] ont fait apparaître l'obésité et le surpoids comme des problèmes grandissants. À ce jour, 40% des français sont concernés par des problèmes de poids : 13 millions sont en état de surpoids et plus de 4 millions sont obèses. Ce phénomène ne se limite pas uniquement à la France (cf. Figure 7.3.1). Le taux d'accroissement de la maladie est alarmant autant en Europe que dans le reste du monde : celui-ci varie de 16% pour la France, 70% pour les États-Unis et monte jusqu'à 200% en Angleterre. Les causes de l'obésité sont nombreuses et peuvent être liées aux habitudes de vies (arrêt de la cigarette, habitudes alimentaires), à des dérèglements hormonaux ou encore à son caractère héréditaire. Par ailleurs, la prise de poids peut entraîner divers problèmes :

- Physiques (diabète, l'hypertension et l'apnée du sommeil) ;
- Psychologiques (dépression, mésestime de soi,…) ;
- Sociaux (discrimination, isolement ...).

L'obésité altérant fortement l'état de santé, elle est désormais considérée comme une maladie. Cependant, aucune thérapie n'est actuellement totalement satisfaisante. Les raisons de ces échecs sont notamment l'aspect pluridisciplinaire de la maladie, demandant un suivi à la fois psychologique, nutritionnel, que culturel et social, voire socio-économique. Impliquant de

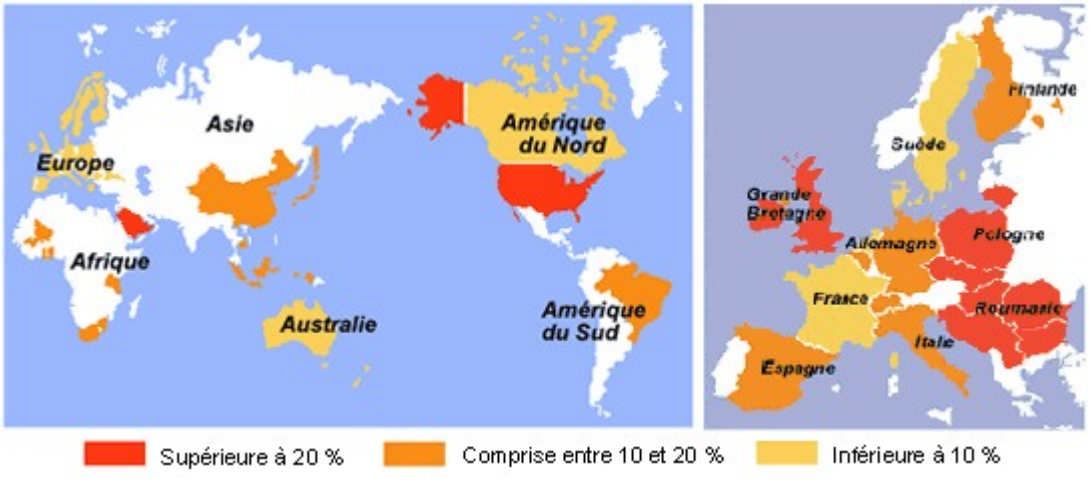


Figure 7.3.1 - L'obésité dans le monde et en Europe

multiples médecins, les interventions mises en oeuvre sont souvent désynchronisées et non coordonnées. L'aspect pluridisciplinaire est également à l'origine d'un manque de connaissances certain de la maladie, notamment des relations existantes entre les disciplines médicales.

Par ailleurs, la croissance du nombre d'ordinateurs, l'émergence de technologies mobiles communicantes (PDA, téléphones mobiles, ...) et la pénétration d'Internet proposent une nouvelle infrastructure matérielle et logicielle. Cette infrastructure facilite la création et l'adoption de nouveaux usages mobiles et ubiquiste. Cette évolution se répercute naturellement dans le domaine de la santé avec le développement de la télé-médecine et de services de santé médiatisés par l'informatique. Ces nouveaux services de santé sont regroupés sous le terme « e-health ». L'exemple le plus populaire actuellement est certainement le Dossier Médical Personnel[8]. Au sein de ce domaine, de plus en plus d'applications délaissent un point de vue « institutionnel » des services de santé pour proposer des applications « citoyennes » (p-health pour personal health). De telles applications permettent aux utilisateurs d'accéder à leurs données personnelles mais également, d'interroger des services de santé dès qu'ils ont en besoin. L'idée est de responsabiliser les utilisateurs en les mettant en situation.

Afin de responsabiliser les personnes obèses, nous proposons une approche citoyenne : un assistant logiciel personnalisé et ubiquiste. Dans la section suivante, nous décrivons la

[8]http://www.d-m-p.org/

démarche réalisée pour acquérir des données, extraire la connaissance et personnaliser l'application.

7.3.1.2 Démarche

L'assistant logiciel, appelé Compagnon, assiste un utilisateur dans le suivi de sa thérapie. Le Compagnon propose une série de conseils appropriés au comportement (également appelé typologie) de son utilisateur. Les conseils sont fournis par les experts médicaux puis attribués à une ou plusieurs typologies. Ces conseils sont caractérisés par un ensemble de conditions. Le(s) conseil(s) dont les conditions sont remplis seront prodigués à l'utilisateur.

Pour déterminer les conseils à proposer à l'utilisateur, le Compagnon doit déterminer la typologie de ce dernier. Pour cela, il utilise les connaissances de trois disciplines médicales : la nutrition, la psychologie et l'activité physique. Ces disciplines prennent une part importante dans l'obésité. Nous avons notamment mentionné les effets négatifs de l'obésité sur les caractères physique et psychologique des personnes obèses.

Néanmoins, ces connaissances sont insuffisantes. Pour déterminer le comportement de son utilisateur, le Compagnon a également besoin de récupérer certaines données concernant ce dernier. Ces données sont récupérées par le biais de questionnaires régulièrement remplis par l'utilisateur. Ces questionnaires sont également un moyen de récupérer un ensemble de données conséquent pour extraire de nouvelles connaissances sur l'obésité et pallier ainsi le manque de connaissances des experts.

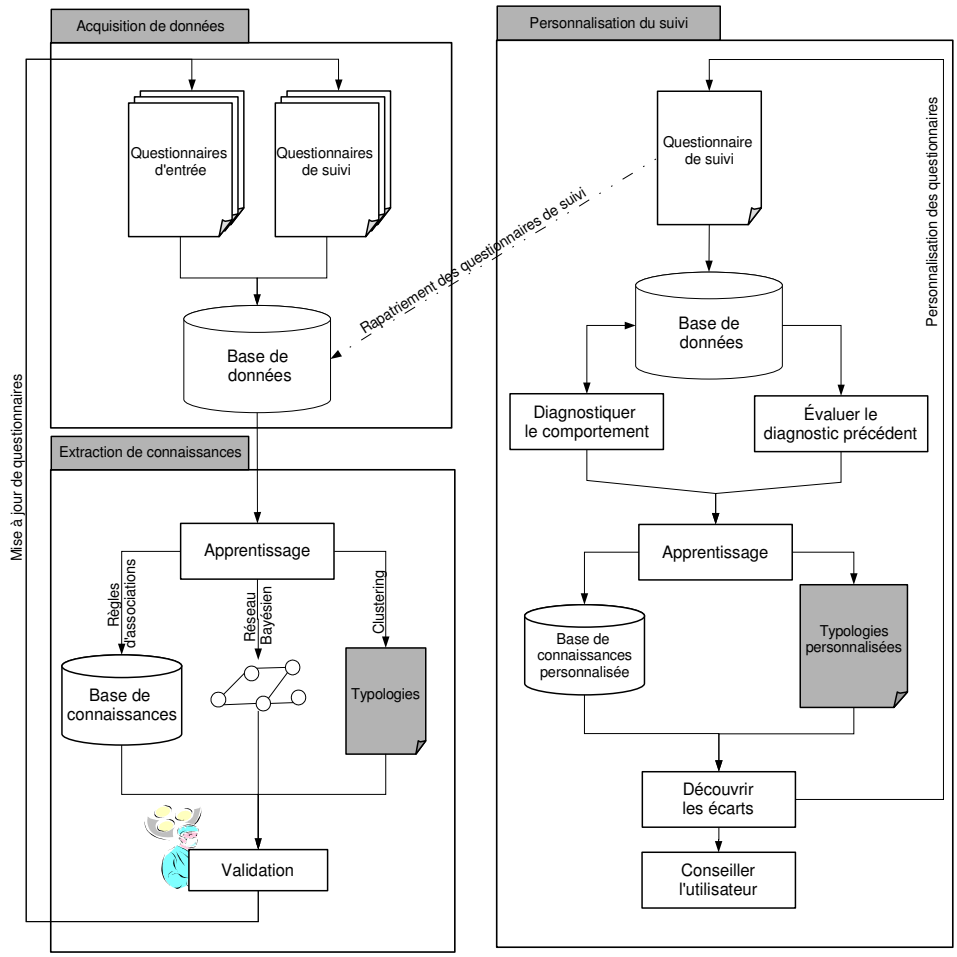


Figure 7.3.2 - Fonctionnement de l'application de Télé-consultation de l'obésité

Dans cette section, nous détaillons la démarche réalisée pour acquérir les connaissances nécessaires au diagnostic de l'utilisateur. Pour cela, nous envisageons une phase d'acquisition de données dans laquelle nous introduisons les questionnaires (cf. section 7.3.1.3). Cette phase d'acquisition de données est suivie d'une phase d'extraction de connaissances (cf. section 7.3.1.3.1) dont le but est de découvrir de nouvelles connaissances à l'aide de techniques d'apprentissage. Puis, nous présentons brièvement la phase de personnalisation des questionnaires (cf. section 7.3.1.3.2) qui permet l'obtention d'une application propre à chaque utilisateur. Cette démarche est schématisée dans la Figure 7.3.2.

7.3.1.3 Acquisition de données

L'obésité est peu connue en tant que maladie pluridisciplinaire. Néanmoins, les experts médicaux sont capables d'en déterminer les causes en utilisant des connaissances personnelles, des connaissances partagées (i.e., des connaissances inhérentes à la discipline de l'expert) et

certaines données relatives aux patients. Les données d'un patient sont les résultats des examens pratiqués par l'expert médical sur le patient. L'obésité étant le résultat de plusieurs facteurs, il existe des relations entre les données de disciplines distinctes. Pour identifier le comportement du patient (également appelé typologie), il est alors nécessaire de connaître ces relations.

Ces relations sont généralement inconnues des experts médicaux car ces derniers sont spécialisés dans une discipline unique. Pour cela, nous construisons un ensemble de questionnaires couvrant les trois domaines médicaux considérés pour découvrir ces relations à l'aide de méthodes d'apprentissage (cf. section 7.3.1.3.1). Les données de ces questionnaires sont ensuite utilisées comme ensemble d'apprentissage. Deux types de questionnaires ont été définis :

1. Questionnaires d'entrée : A chaque domaine est associé un questionnaire type permettant de cibler la typologie initiale du patient. Ces questionnaires, appelés *questionnaires d'entrée*, comportent des séries de questions révélatrices des typologies connues de l'obésité. Les connaissances relatives à ces typologies permettent de fournir au Compagnon une base de connaissances représentative du comportement initial de son patient. Cette base de connaissances est ensuite personnalisée à l'aide des questionnaires de suivi.
2. Questionnaires de suivi : Les questionnaires de suivi ont trois fonctions. La première fonction est relative au caractère citoyen de l'application et vise à responsabiliser les patients au travers de questions sensibles. De telles questions mettent en évidence les écarts réalisés par les patients au cours de leur thérapie. Leur seconde fonction est l'interprétation et la détermination du comportement courant du patient. À cet effet, les questionnaires de suivi sont composés de questions choisies en fonction du comportement précédemment diagnostiqué par le Compagnon. Les connaissances apportées par les réponses à ces questions permettent de personnaliser (cf. section 7.3.1.3.2) la base de connaissances du Compagnon. Enfin, les questionnaires de suivi contiennent un ensemble de questions permettant de calculer la pertinence des conseils proposés précédemment.

Le questionnaire d'entrée est rempli par le patient à son entrée dans le système. Il est ensuite proposé occasionnellement aux patients pour confirmer l'évolution de leur comportement. Contrairement aux questionnaires de suivi, il tend à attribuer une typologie précise aux patients. De ce fait, il contient un nombre important de questions.

Les questionnaires de suivi sont construits par le Compagnon (cf. section 7.3.1.3.2) qui, pour répondre au mieux aux critères d'une application p-health est déployé sur un petit objet portable (PDA, téléphone, ...). Pour que l'assistant soit ergonomique, les questionnaires de suivi sont relativement courts et contiennent au maximum une trentaine de questions. Ils ont l'avantage d'être peu contraignants car rapidement remplis par le patient. Les questionnaires présents sur le Compagnon sont régulièrement rapatriés au niveau d'un serveur de données. Les réponses sont alors enregistrées dans des bases de données dont les attributs identifient les questions présentes soit dans le questionnaire d'entrée soit dans un questionnaire de suivi (cf. Figure 7.3.2, *Acquisition de données*).

Afin d'avoir suffisamment de données pour extraire de la connaissance, nous souhaitons suivre une cohorte de cinq cents personnes pendant quatre mois. En plus d'un questionnaire d'entrée, les personnes ainsi suivies remplissent, sur papier ou Internet, deux à trois questionnaires de suivi par semaine. À cause du manque de connaissances sur l'obésité, la personnalisation des questionnaires (cf. section 7.3.1.3.2) n'est pas envisagée durant cette phase d'acquisition. Par conséquent, les questionnaires de suivi sont identiques pour chaque personne de la cohorte.

7.3.1.3.1 Extraction de connaissances

Les experts médicaux ne possèdent pas les connaissances nécessaires pour décrire l'obésité sur les trois disciplines considérées. Alors, nous utilisons des techniques d'apprentissage pour extraire ces connaissances des données récoltées pendant la phase d'acquisition. Les techniques d'apprentissage choisies sont le clustering, les réseaux Bayésiens et les règles d'associations (cf. Figure 7.3.2, *Extraction de connaissances*).

1. **Le clustering** : le clustering est une méthode d'apprentissage non supervisée dont le but est de regrouper en clusters (ou ensembles) des individus sensiblement similaires. Dans notre cas, nous l'utilisons pour découvrir les typologies des personnes atteintes d'obésité. Ces typologies sont définies à l'aide des similarités entre les patients d'un même cluster. Par ailleurs, le clustering permet d'affiner les typologies définies initialement par les experts médicaux. À cause du manque de connaissances des experts, ces typologies attribuées par les questionnaires d'entrée, sont probablement peu pertinentes. Le clustering permet d'affiner ces typologies et de mieux cibler les comportements types des patients. De plus, elles sont susceptibles d'être invalidées par les typologies nouvellement découvertes par clustering.
2. **Un réseaux Bayésiens** : un réseaux Bayésien est utilisé pour attribuer une typologie aux patients même en l'absence d'une ou plusieurs informations (tel est le cas lorsque le

Compagnon s'appuie sur les réponses d'un questionnaire de suivi). Ce réseau Bayésien est construit sur l'intégralité des données récoltées et permet de découvrir les relations existantes entre la nutrition, la psychologie et l'activité physique. Le réseau Bayésien ainsi appris nous permet de considérer les relations inter-domaines lors du diagnostic.

3. **Les règles d'associations** : les bases de données présentes sur le serveur sont utilisées pour construire les bases de connaissances du Compagnon. Ces bases de connaissances contiennent les règles d'associations, préalablement validées par les experts médicaux, qui décrivent de manière pertinente les trois disciplines (nutrition, psychologie et activité physique). Les experts médicaux ont également la possibilité d'utiliser leur expérience pour construire leurs propres règles et les ajouter à la base de connaissance du Compagnon. Par la suite, le Compagnon se sert des règles contenues dans sa base de connaissances pour déduire la valeur de certains noeud du réseau Bayésien (cf. Chapitre 6).

7.3.1.3.2 Personnalisation des questionnaires

Afin de personnaliser le suivi (cf. Figure 7.3.2, *Personnalisation du suivi*), les bases de connaissances du Compagnon s'adaptent à son utilisateur. Pour cela, de nouvelles règles d'associations sont produites à partir des réponses aux questionnaires de suivi. D'autres sont supprimées car elle ne correspondent plus aux données des patients. Par ailleurs, des typologies personnalisées peuvent également être identifiées grâce aux comportements diagnostiqués par le Compagnon. Le processus utilisé pour acquérir ses nouvelles connaissances est similaire à celui décrit dans la section 7.3.1.3.

La personnalisation des questionnaires permet ainsi de i) valider les règles produites, ii) valider les typologies personnalisées et iii) de cibler plus précisément le comportement de l'utilisateur en fonction du diagnostic précédemment réalisé par le Compagnon. Pour cela, le Compagnon choisit parmi les questions disponibles, celles qui lui permettent de corroborer ses opinions (cf. section 7.3.1.3).

7.3.1.4 Architecture du Compagnon Santé

Embarqués dans des objets portables (POP) tels que PDA ou téléphones mobiles 3G « smart phones », les Compagnons (cf. Figure 7.3.3) proposent une aide et une assistance personnalisées aux personnes en état de surpoids, dans leur quotidien. Les Compagnons sont insérés au sein d'un réseau de télé-santé (réseau compagnon santé) intégrant un serveur distant (cf. Figure 7.3.3). Ce serveur centralise les données collectées par les Compagnons auprès de leur utilisateur. Il traite et analyse régulièrement cet ensemble de données pour en extraire des connaissances sur l'obésité et ses dimensions physique, psychologiques et nutritionnelles. Le

serveur est également le point d'entrée des experts médicaux participant au réseau de télé-santé. Ces derniers peuvent i) consulter les historiques des différents utilisateurs, ii) interagir avec un utilisateur via son Compagnon et iii) interagir avec d'autres experts du système.

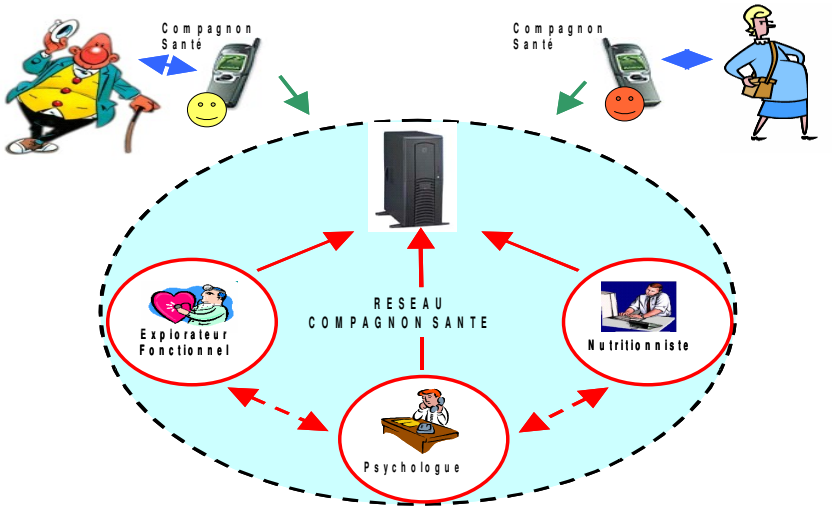


Figure 7.3.3 - Architecture du réseau Compagnon Santé

Le déploiement des Compagnons sur des POP est motivé par la volonté d'obtenir un suivi continu répondant au mieux à la définition d'une application p-health. Bien que les capacités des POP soit en constantes évolutions (autonomie, mémoire, ...), ces derniers restent moins performants qu'un ordinateur. C'est pour cette raison que nous déléguons le diagnostic à un ensembles d'agents composé uniquement de cinq agents (cf. Figure 7.3.4). Trois de ces agents représentent les experts médicaux : l'Agent Psychologue, l'Agent Nutritionniste et l'Agent Physicien. Ces Agents de domaine interagissent avec un Agent Coordinateur qui joue à la fois le rôle d'Agent Bayésien et d'Agent Superviseur (cf. Chapitre 5 et Chapitre 6). La définition d'un tel agent est justifiée par la présence d'un unique Agent Expert par discipline.

Afin de calculer leur diagnostic, les Agents de domaine ont besoin des bases de connaissances correspondant à leur discipline et des réponses au questionnaire de suivi. Ces réponses sont récupérées par l'Agent d'Interface qui les transmets à l'Agent Coordinateur (cf. Chapitre 6). L'Agent d'Interface gère les interactions avec l'utilisateur en i) proposant les questionnaires de suivi, ii) affichant le diagnostic des Agents du domaine et iii) affichant les conseils adaptés au diagnostic.

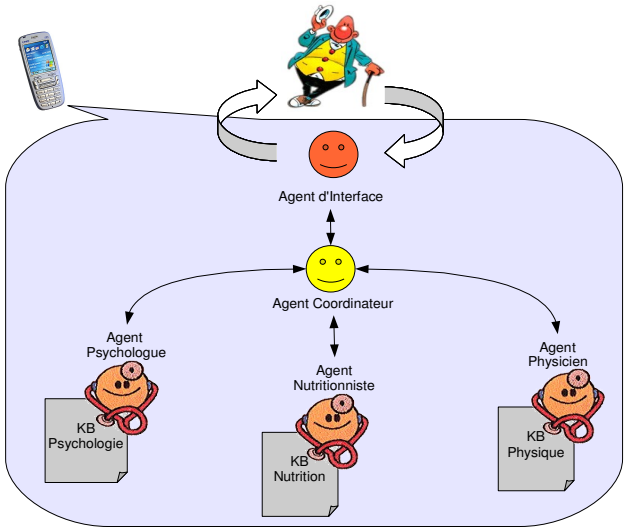


Figure 7.3.4 - Architecture du Compagnon Santé

7.3.2 Laminoir circulaire

La mise en place du réseau Compagnon Santé demandant trop d'efforts pour être réalisée dans la durée de la thèse, nous avons appliqué notre modèle à un problème mécanique: le laminoir circulaire. Le laminoir, circulaire que nous considérons, est une machine d'origine allemande permettant de réaliser des pièces métalliques circulaires de grand diamètre. Le coût de laminage de telles pièces est élevé et demande un excédent de matière important, pouvant atteindre 50% du poids de la couronne finie. Par conséquent, l'apparition de défauts sur les pièces entraîne des pertes importantes qu'il faut minimiser au maximum. Pour cela, il est nécessaire d'identifier l'origine de ces défauts. Le fonctionnement du laminoir est détaillé dans la section 7.3.2.1.

Le problème fondamental du laminoir circulaire provient d'un manque de connaissances du procédé de laminage. Bien que les utilisateurs aient acquis le savoir-faire nécessaire pour réduire les défauts (cf. section 7.3.2.1), ils n'en connaissent pas la ou les origines. Connaître les origines d'un défaut est cependant essentiel pour éviter leur réapparition. Nous utilisons les données d'usinage des pièces pour découvrir l'origine des défauts. Nous présentons la démarche appliquée pour mettre en oeuvre le modèle dans la section 7.3.2.2. Dans la section Erreur : source de la référence non trouvée, nous présentons les résultats obtenus par l'application de notre modèle au problème du laminoir circulaire.

7.3.2.1 Contexte et motivations

La construction de pièces circulaires repose initialement sur le cintrage de barres de fer sous forme de couronne. Depuis, le procédé de laminage a évolué et est délégué à une installation de laminage circulaire appelée laminoir circulaire (ou laminoir radial/axial). Une couronne est désormais le résultat d'une déformation consistant à l'élargissement par laminage d'une ébauche circulaire perforée, dont le diamètre intérieur est le plus petit possible, jusqu'à atteindre les dimensions finales souhaitées. Pour passer de la taille d'ébauche à la taille finale, la couronne subit une déformation simultanée de son épaisseur et de sa hauteur (cf. Figure 1.4.1). La réduction de l'épaisseur (resp. la hauteur) se fait par écrasement de la pièce entre le cylindre principal et le mandrin (resp entre le cône inférieur et le cône supérieur).

Un laminoir circulaire est constitué de cinq parties (cf. Figure 7.2.3) : un bâti fixe sur lequel est apposé le cylindre principal, la cage radiale (mobile) avec le mandrin, la cage axiale (également mobile) avec les cônes, le chariot axial logé dans la cage axiale portant le cône supérieur et les bras de centrage servant à maintenir la couronne dans l'axe longitudinal du laminoir. La cage axiale se déplace vers l'arrière pour accompagner la pièce. En parallèle, le chariot axial et la cage radiale sur lesquels sont respectivement fixés le cône supérieur et le mandrin se déplacent de manière à contrôler l'épaisseur et la hauteur de la couronne.

La réalisation d'une couronne exige un excédent de matière dans le but de pallier le manque de précision du processus de laminage. Cet excédent, pouvant atteindre 50% du poids de la couronne finie, est retiré durant la phase d'usinage par enlèvement de copeaux. L'importance de cet excédent s'explique par les défauts de forme qui peuvent se présenter après le laminage. Les défauts de laminage identifiés sont les suivants :

- Rétractations au niveau des faces des couronnes (cf. Figure 7.3.5a);
- Incurvations des couronnes dues à la remontée de la couronne contre le cylindre principal (cf. Figure 7.3.5b);
- Voilage axial, particulièrement pour les couronnes de grand diamètre (cf. Figure 7.3.5c);
- Ovalisation de la couronne (cf. Figure 7.3.5d);
- Défaut de calibrage : une ou plusieurs dimensions (épaisseur, hauteur, diamètre intérieur ou extérieur) ne correspondent pas à celles définies dans la recette;
- Écrasement de la pièce par les bras de centrage (cf. Figure 7.3.5e).

Grâce à l'automatisation du processus de laminage, de tels défauts sont peu courants et résultent en général d'un dysfonctionnement du laminoir. Les défauts occasionnés par un dysfonctionnement (mandrin voilé, problème hydraulique, etc) sont récurrents. Par conséquent, leur origine est aisément identifiable. Pourtant, il arrive que certains défauts soient occasionnels. Dans un tel cas, l'origine n'est pas mécanique mais provient d'une ou plusieurs caractéristiques de la couronne (température de l'ébauche insuffisante, perforation de l'ébauche non centrée, etc).

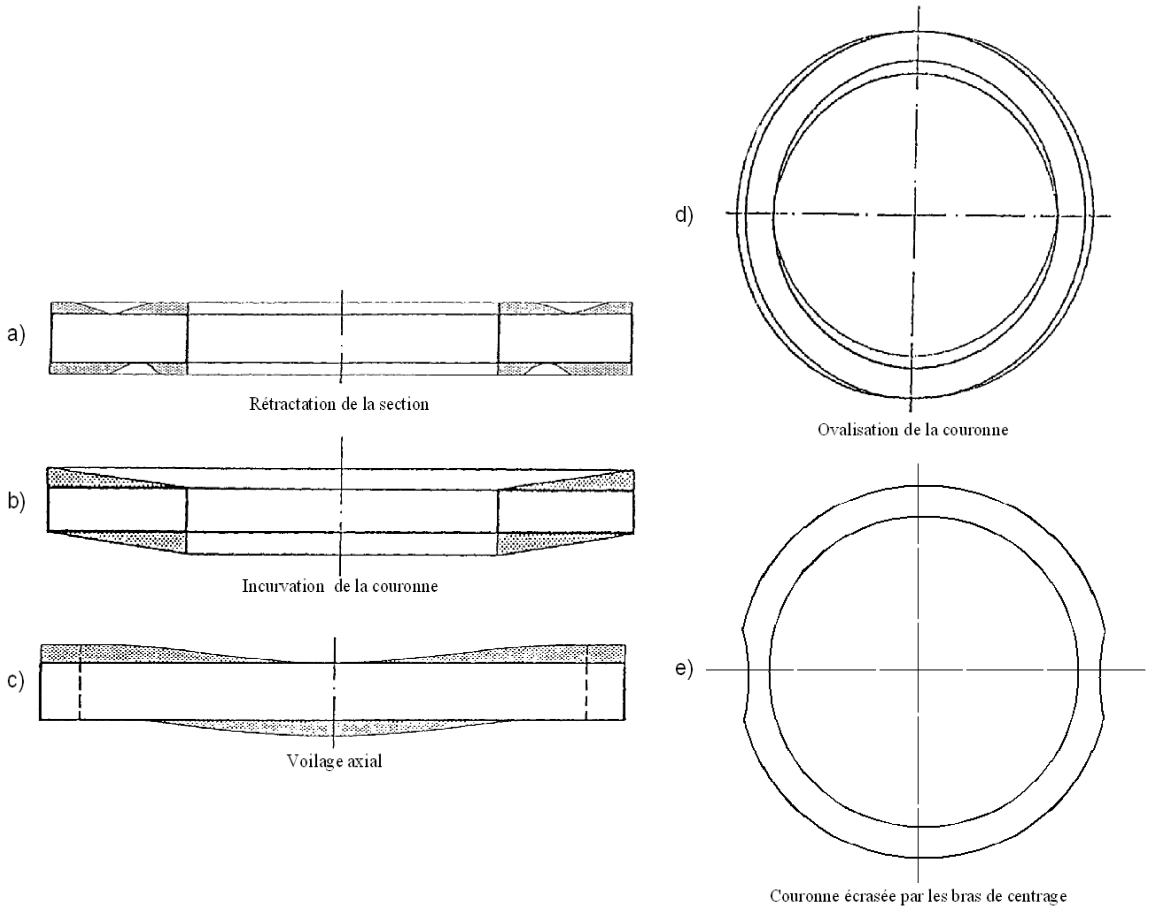


Figure 7.3.5 - Défauts occurrents pendant le laminage circulaire

Dans chacun des cas précédemment cités, les utilisateurs du laminoir possèdent le savoir-faire nécessaire pour laminer des couronnes, mais pas pour diagnostiquer l'origine des défauts. Les coûts de laminage étant élevés, notamment à cause de l'excédent de matière, il est important de minimiser les pertes. C'est pour cette raison qu'un diagnostic rapide du laminoir est requis. Nous déléguons ce diagnostic au système multi-agents que nous décrivons dans la section suivante.

7.3.2.2 Démarche

Les forges dans lesquelles sont installés les laminoirs usinent les couronnes 24h/24. Durant la journée, trois équipes de lamineurs se relaient. Chacune de ces équipes possède une expérience personnelle du laminage circulaire. Pour minimiser les pertes, cette expérience est utilisée pour répartir les couronnes en fonction de leur difficulté. La difficulté d'une couronne réside dans ses caractéristiques (épaisseur, diamètre, matière, etc) appelées recette. Une couronne de large diamètre est, par exemple, plus difficile à réaliser parce qu'elle peut se voiler durant le procédé de laminage.

De nos jours, le procédé de laminage est automatisé. Les lamineurs peuvent néanmoins, à l'aide de potentiomètres, ajuster manuellement les paramètres du laminoir (la pression des bras de centrage, la vitesse de laminage et la vitesse de croissance) afin d'éviter l'apparition de certains défauts. Pour cela, ils mettent à contribution leur savoir-faire et leur expérience. Les lamineurs connaissent la solution adaptée à un défaut mais sont incapables d'en déterminer l'origine. Dans ce but, nous utilisons les données enregistrées au cours du laminage.

Les données d'usinage d'une couronne sont conservées dans une base de données gérées par l'automate. Cette base est appelée « base de process ». Au cours du laminage, l'automate enregistre toutes les 500ms, l'état courant de la couronne dans la base de process. Chaque enregistrement est constitué de 37 attributs qui représentent : l'identifiant de la couronne, ses dimensions (hauteur, épaisseur, diamètre, etc), les forces exercées par le laminoir (force axiale, force radiale, etc) et une recette partielle de la présente couronne (matière, taille d'ébauche, équipe, etc). Les ajustements effectués par les lamineurs sont également contenus dans la base de process et correspondent à trois des 37 attributs mentionnés précédemment. Par ailleurs, les défauts sont manuellement enregistrés en fin de laminage par un membre de l'équipe. Ces défauts sont sauvegardés dans une base, appelée « base utilisateur », dans laquelle ils sont référencés par l'identifiant de la pièce défectueuse. Ces deux bases nous servent à découvrir les connaissances nécessaires à la réalisation des agents.

Le modèle proposé (cf. Chapitre 5) permet de prendre en compte l'expérience d'une communauté d'experts appartenant à des domaines d'expertise distincts. Dans le cas du laminoir, nous avons choisi de partager le problème en quatre domaines. Trois de ces domaines sont définis suivant la structure du laminoir : i) la cage axiale, ii) la cage radiale et iii) l'ensemble composé des bras de centrage et du chariot axial. Le dernier domaine considère uniquement les données relatives aux recettes des couronnes (dimensions d'ébauche, dimensions finales et matière). Ces domaines sont composés de trois experts représentant les équipes de lamineurs. Les bases de connaissances des experts sont constituées de règles d'associations apprises sur les données relatives au domaine et à l'équipe considérés. Un réseau

Bayésien est également appris sur l'intégralité de ces données (cf. Chapitre 6). Les règles d'associations et le réseau Bayésien (cf. Figure 7.3.6) ainsi appris sont illustrés dans les figures suivantes.

Le cadre ci-dessous contient un échantillon des règles d'associations apprises à partir des données du laminoir. Une règle : $attribut_1 = (v_{11}\text{-}v_{12}] \Longrightarrow attribut_2 = (v_{21}\text{-}v_{22}]$ signifie que l'$attribut_2$ prend une valeur appartient à $]v_{11}\text{-}v_{12}]$ si la valeur de l' $attribut_1$ est comprise une dans l'ensemble $]v_{21}\text{-}v_{22}]$

Exemple de règles d'associtations

1. coeff_lam='(0.932-1]' ==> choix_arret=30
2. coeff_lam='(0.932-1]' matiere=1 ==> choix_arret=30
3. coeff_lam='(0.932-1]' deltaEpaisseur='(-1.050885-0.987095]' ==> choix_arret=30
4. coeff_lam='(0.932-1]' deltaEpaisseur='(-1.0508-0.9870]' matiere=1 ==> choix_arret=30
5. diametre_ini='(648.641211-728.239014]' ==> choix_arret=30
6. dec_ang_bras='(-0.506399-0.886401]' coeff_lam='(0.932-1.004]' ==> choix_arret=30
7. f_radiale='(208.6585-288.3389]' choix_arret=30==> deltaEpaisseur='(-1.050885-0.9870]'
8. diametre_ini='(489.445605-569.04340]' deltaEpaisseur='(-1.0508-0.9870]' ==> matiere=1

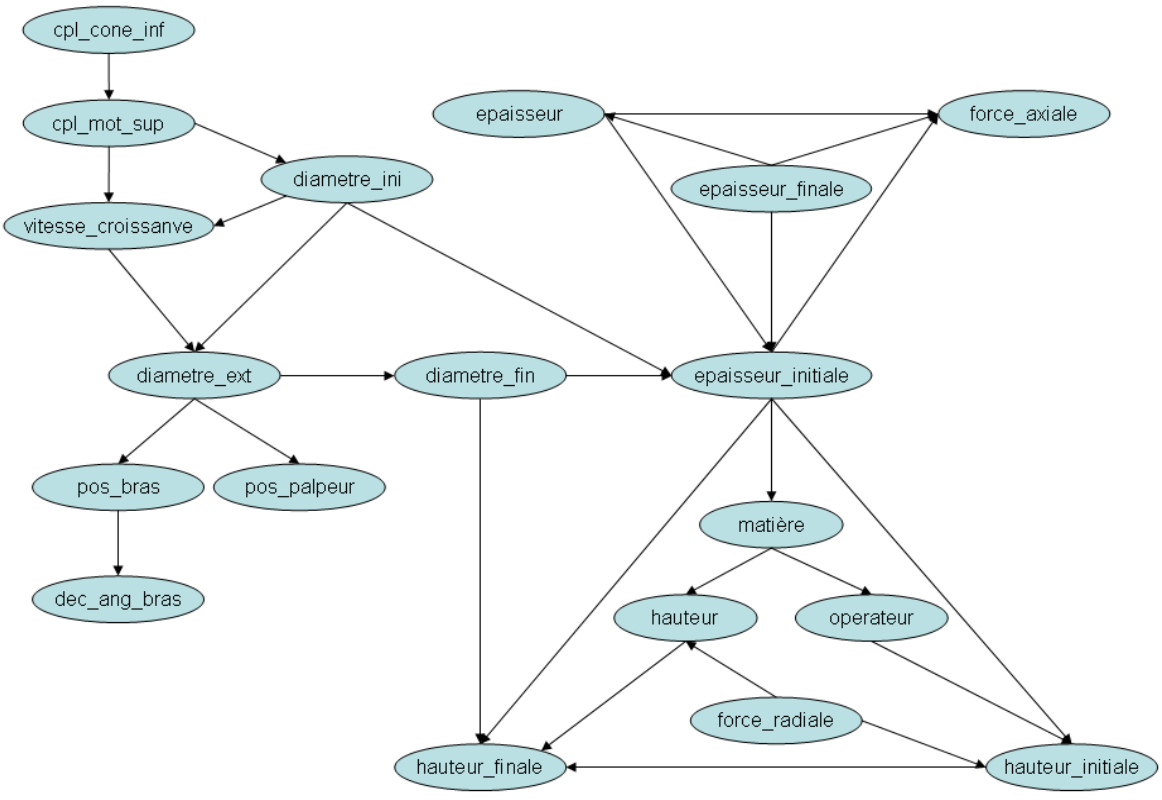


Figure 7.3.6 - Réseau Bayésien appris sur les données du laminoir

7.3.2.3 Architecture de l'outil de diagnostic

Les quatre domaines précédemment cités sont tous composés de trois Agents de domaines (cf. Figure 7.3.7). Ces Agents de domaines représentent les trois équipes de lamineurs. Leurs bases de connaissances sont constituées des règles d'associations apprises sur les données relatives au domaine et à l'équipe considérés. Par exemple, l'Agent du domaine en *Recettes* nommé « Équipe 1 » possède uniquement, des règles d'associations apprises sur les données des recettes des couronnes usinées, par la première équipe de lamineurs.

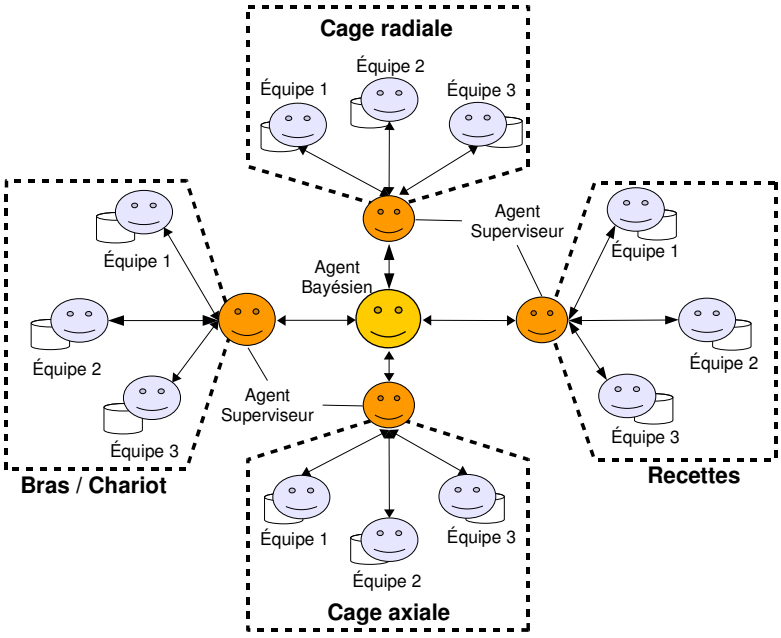


Figure 7.3.7 - Architecture de l'outil de diagnostic

Les Agents Superviseur sont contactés par l'Agent Bayésien (cf. Figure 7.3.7) dans le but d'obtenir les informations supplémentaires (cf. Chapitre 5). L'Agent Bayésien permet aux membres d'une équipe de rentrer les caractéristiques de la pièce défectueuse mais également, de leur présenter le diagnostic réalisé conjointement avec les Agents de domaine et les Agents Superviseur.

7.4 Synthèse

Dans ce chapitre, nous avons présenté les outils de diagnostic que nous avons réalisé à l'aide du modèle agent proposé dans les Chapitre 4 et 5. Cet outil permet d'élaborer un diagnostic à l'aide des connaissances extraites des données ou fournies par les experts. Le diagnostic ainsi réalisé est basé sur i) des arguments créés et évalués par les agents du système

et ii) le réseau Bayésien décrivant les relations de cause à effet entre les attributs du domaine. Afin d'expliquer le diagnostic, nous reformulons les arguments et les probabilités du réseau.

Souhaitant gardé l'expertise des experts intacte, nous avons attribué un Agent du domaine à chacun des experts humains du système. Ces agents possèdent les connaissances de l'expert auquel ils sont rattachés. Dans ce même but, nous avons opté pour l'utilisation d'un agent centralisateur : l'Agent Superviseur. Cet agent joue, en quelque sorte, le rôle de blackboard : il autorise un Agent du domaine à demander l'avis de ses « confrères ».

De ce fait, le système permet de construire un diagnostic de façon collaborative en se basant sur l'expérience d'une communauté d'experts (issues des données ou de l'expert lui-même). D'autre part, le système fournit à l'utilisateur un ensemble d'arguments qui justifie le diagnostic et lui permet de comprendre l'origine du problème.

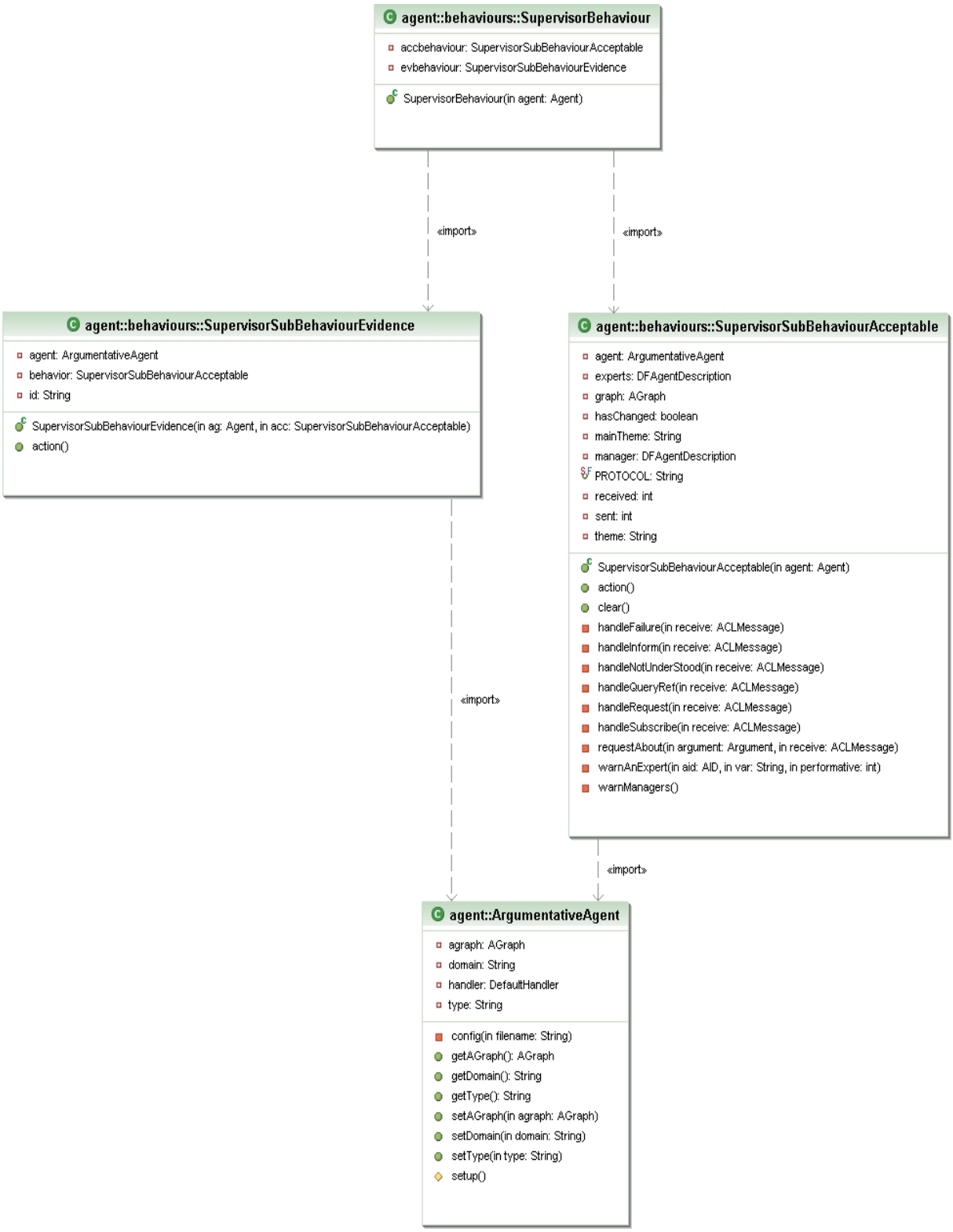


Figure 7.4.1: - Diagramme de classe de l'agent Superviseur

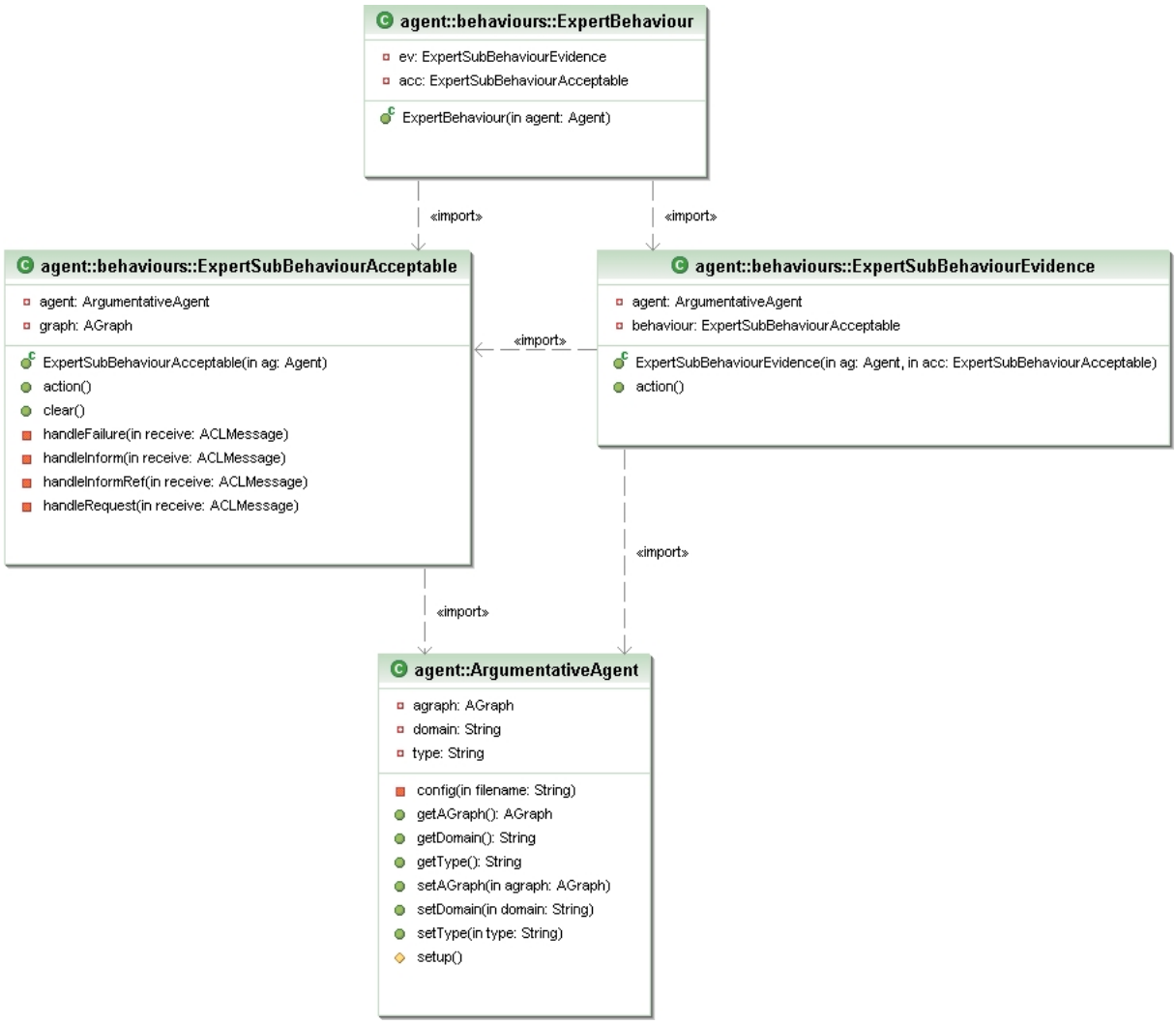


Figure 7.4.2: - Diagramme de classes des Agents de domaine

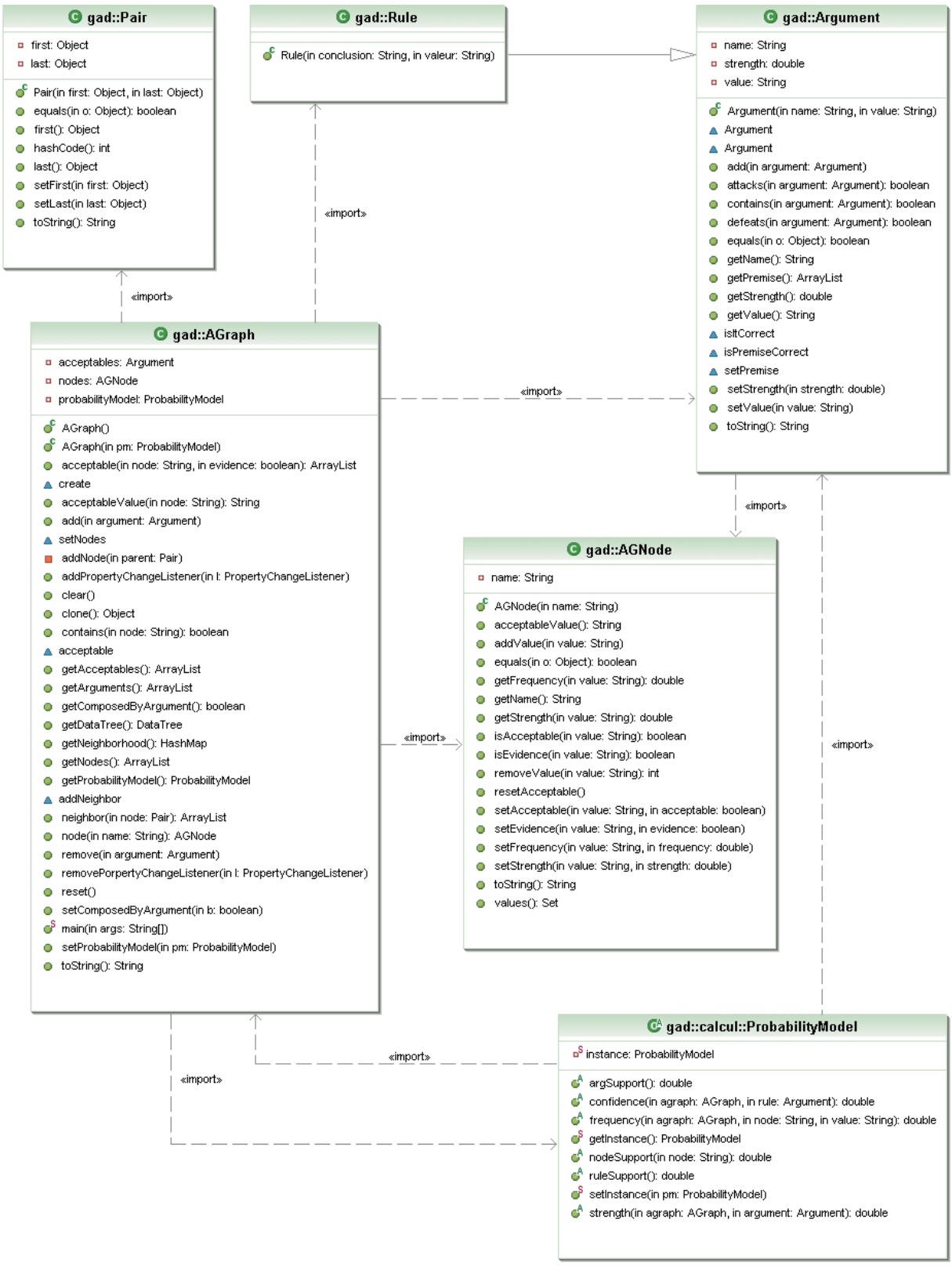


Figure 7.4.3: - Diagramme de classes du graphe argumentatif

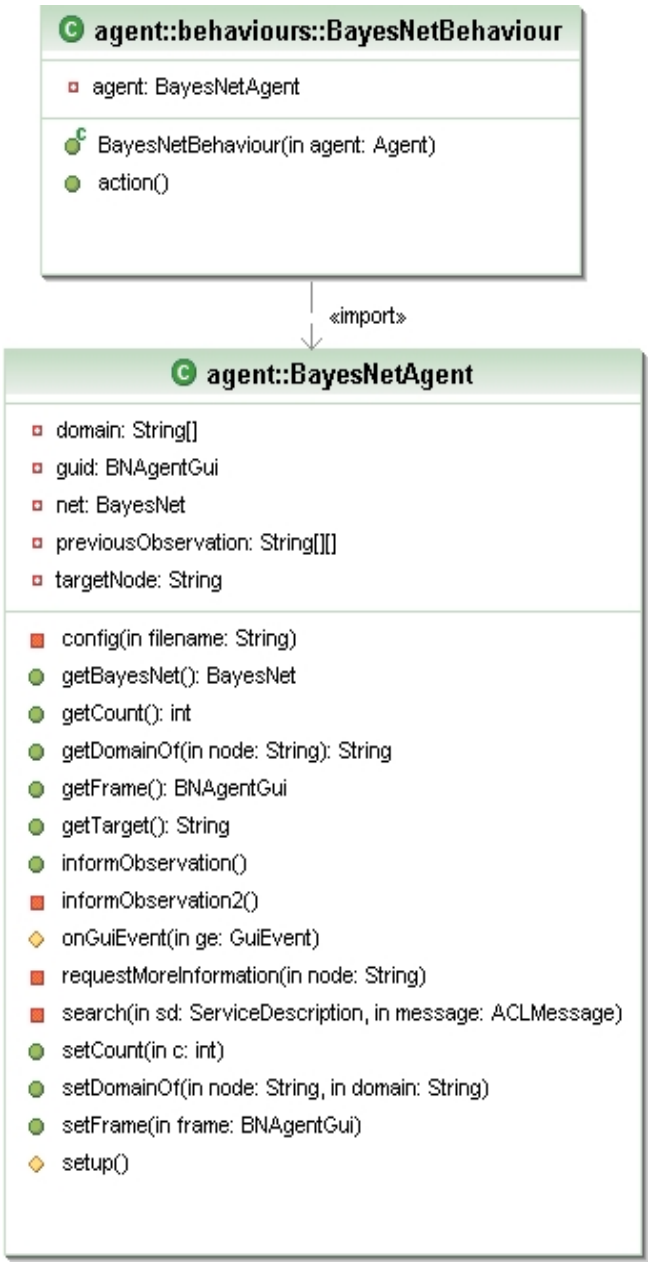


Figure 7.4.4: - Diagramme de classes du comportement de l'Agent Bayésien

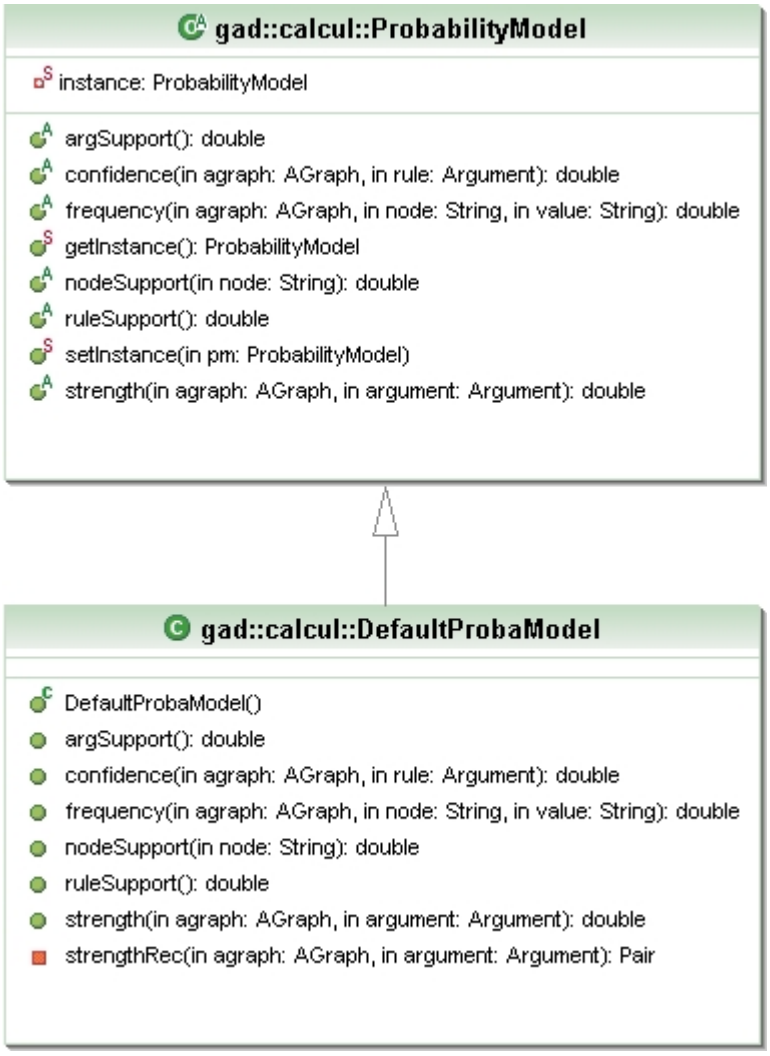


Figure 7.4.5: - Diagramme de classes du modèle argumentatif

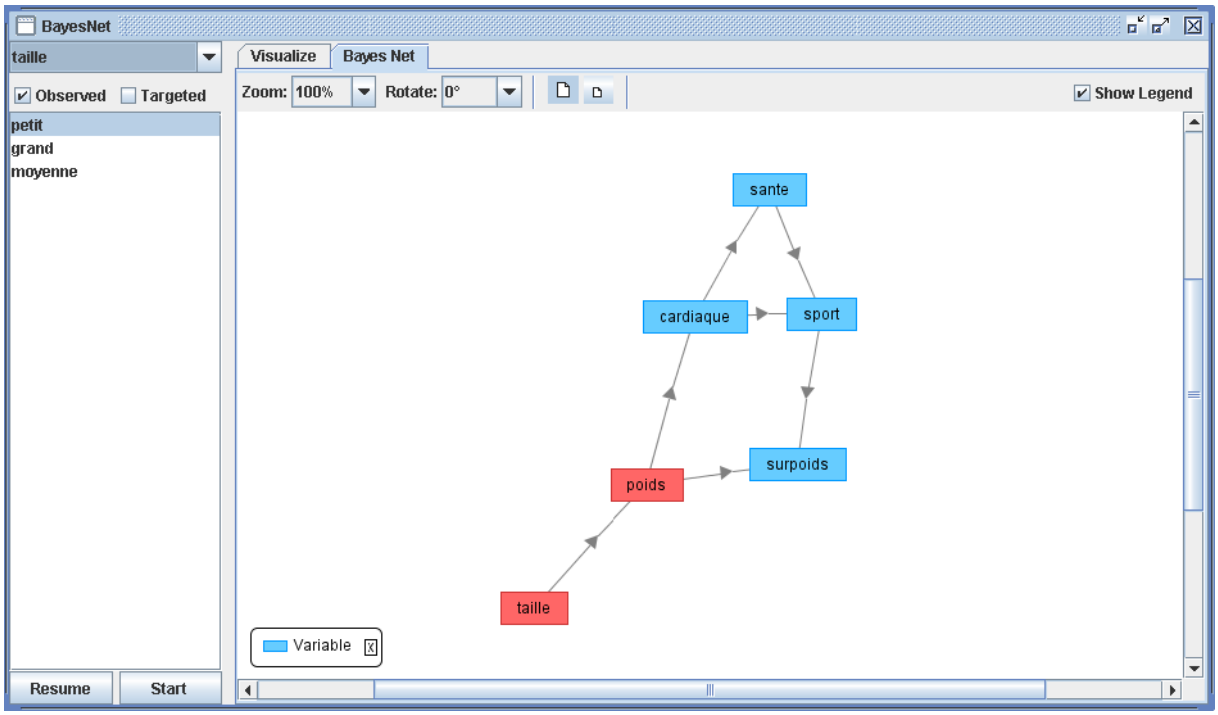


Figure 7.4.6: - Réseau Bayésien géré par l'Agent Bayésien

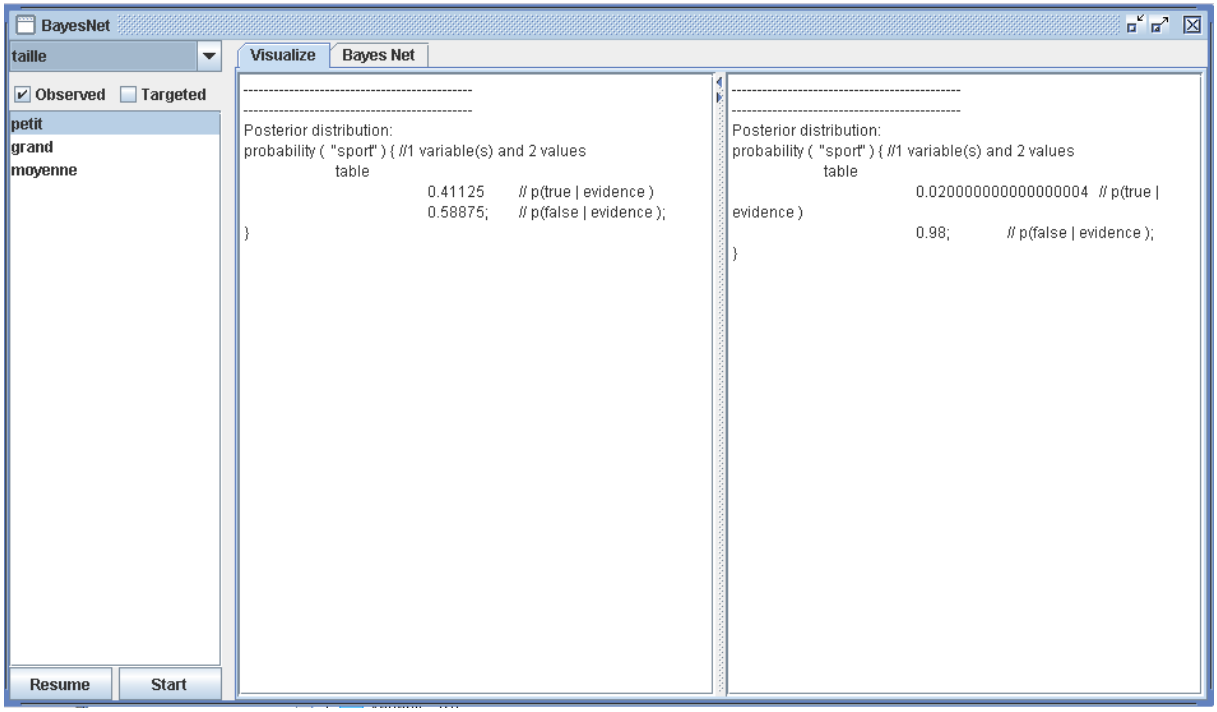


Figure 7.4.7: - Résultats de la propagation des probabilités avant et après argumentation

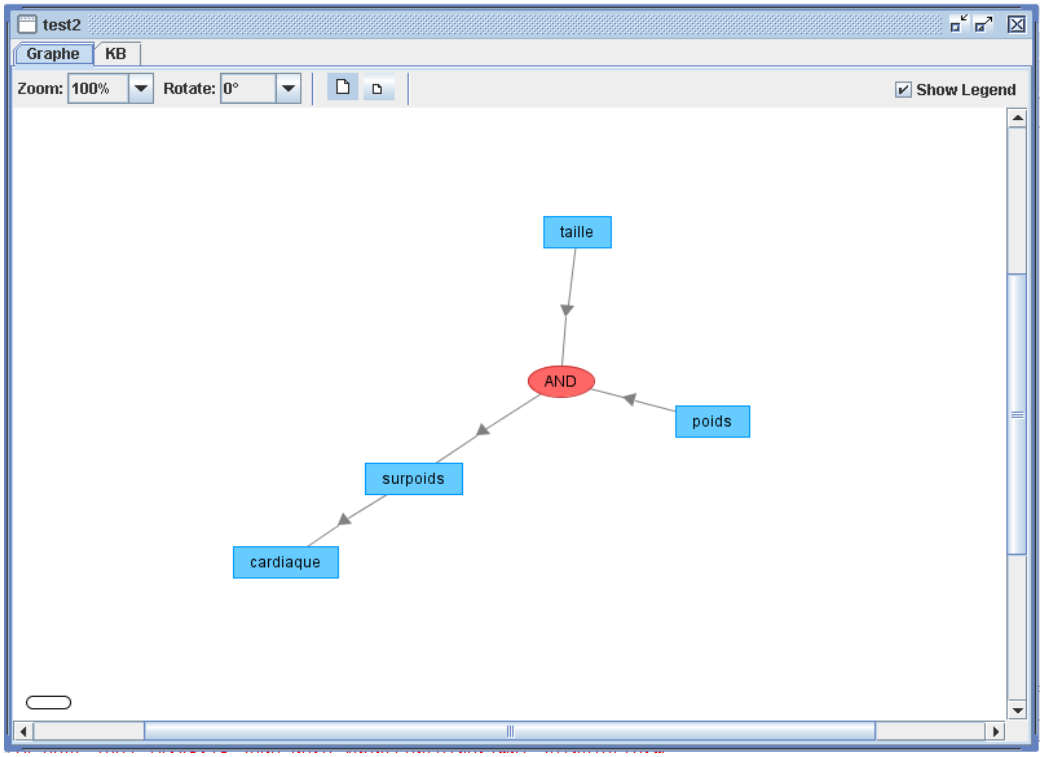


Figure 7.4.8: - Visualisation du graphe argumentif

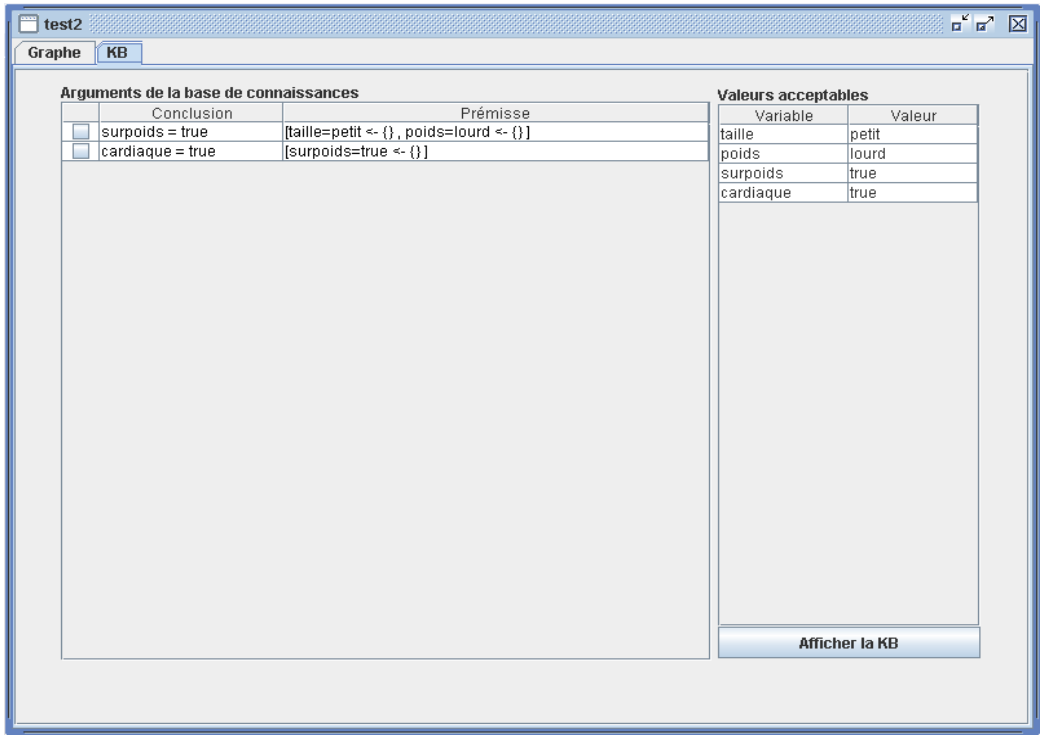


Figure 7.4.9: Visualisation des arguments et des valeurs acceptables

Bibliographie

[1] A. Aamodt and E. Plaza., Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches.Rapport de congrès, 39-52. Artificial Intelligence Communications,1994.

[2] A. Aghasaryan and E. Fabre and A. Benveniste and R. Boubour and C. Jard, A Petri net approach to fault detection and diagnosis in distributed systems.Divers,1997.

[3] R. Agrawal and R. Srikant, Fast Algorithms for Mining Association Rules in Large Databases. 478-499. 20th International Conference on Very Large Data Bases,1994.

[4] D. Alazard, Introduction au Filtre de Kalman. January, 2005.

[5] L. Amgoud and N. Maudet and S. Parsons, Modelling dialogues using argumentation, 2000.

[6] L. Amgoud and C. Cayrol, A Reasoning Model Based on the Production of Acceptable Arguments.197-215. Math. Artif. Intell.1-3, 2002.

[7] L. Amgoud and S. Parsons and N. Maudet, Arguments, Dialogue, and Negotiation.Rapport de congrès, 338-342, ECAI, 2000.

[8] Agent UML Web Site : http://www.auml.org/.

[9] W.M. Bain, Case-based reasoning: a computer model of subjective assessment.New Haven, CT, USA,Yale,Yale University,1986.

[10] P. Baroni and G. Lamperti and P. Pogliano and M. Zanella, Diagnosis of large active systems,.Magazine.135-183,Essex, UK, Artif. Intell., Elsevier Science Publishers Ltd., 1,1999.

[11] T. Bayes, An essay towards solving a problem in the doctrine of chances (1763),.Magazine. 189-215, Bayesian Statistics: Principles, Models, and Applications, 1991.

[12] T. Bayes, An essay towards solving a problem in the doctrine of chances (1763),189-215,Bayesian Statistics: Principles, Models, and Applications,1991.

[13] T.J.M Bench-Capon, Persuasion in Practical Argument Using Value Based Argumentation Frameworks, 429-448, Journal of Logic and Computation, 2003.

[14] T.J.M. Bench-Capon and G. Sartor, A model of legal reasoning with cases incorporating theories and values, 97-143, Artif. Intell., 1-2, 2003.

[15] A. Benveniste and E. Fabre and C. Jard and S. Haar, Diagnosis of Asynchronous Discrete Event Systems: A Net Unfolding Approach, Rapport de congrès, 182,0-7695-1683-1, Washington, DC, USA, IEEE Computer Society, WODES '02: Proceedings of the Sixth International Workshop on Discrete Event Systems (WODES'02),2002.

[16] A. Benveniste and S. Haar and E. Fabre and C. Jard, Distributed Monitoring of Concurrent and Asynchronous Systems, 1-26, CONCUR, 2003.

[17] R.R. Bouckaert, Bayesian Belief Networks: from Construction to Inference, Utrecht, Netherlands,1995.

[18] Ivan Bratko and Ross King, Applications of inductive logic programming, 43-49, New York, NY, USA, SIGART Bull.,ACM Press, 1, 1994.

[19] B.G. Buchanan and E.H. Shortliffe, Rule-based expert systems : the MYCIN experiments of the Stanford Heuristic Programming Project, Mass. : Addison-Wesley, Reading, 1984.

[20] V. Carofiglio, Modelling argumentation with belief networks, In F. Grasso, C. Reed and G. Carenimi. Proc of the 4th Workshop on Computational Models of Natural Argument (CMNA-2004), 2004.

[21] E. Charniak, Bayesian Networks Without Tears, 39-49, AI Magazine,1991.

[22] C.I. Chesnevar and A.G. Maguitman and R.P. Loui, Logical models of argument, 337-383, New York, NY, USA, ACM Comput. Surv.,ACM Press, 4, 2000.

[23] G. Chiola and M. A. Marsan and G. Balbo and G. Conte, Generalized Stochastic Petri Nets: A Definition at the Net Level and its Implications, 89-107,Piscataway, NJ, USA, IEEE Trans. Softw. Eng., IEEE Press, 2, 1993.

[24] CLIPS Homepage : http://www.ghg.net/clips/CLIPS.html.

[25] A. Colmerauer, Prolog II : manuel de référence et modèle théorique, , Marseille : Faculté des Sciences de Luminy, 1982.

[26] T. Cover and P. Hart, Nearest neighbor pattern classification, 21-27, Information Theory, IEEE Transactions, 1967.

[27] R. Debouk and S. Lafortune and D. Teneketzis, Coordinated Decentralized Protocols for Failure Diagnosisof Discrete Event Systems, 33-86, Hingham, MA, USA, Discrete Event Dynamic Systems,Kluwer Academic Publishers, 1-2, 2000.

[28] L. Dehaspe and L. De Raedt and W. Laer, CLAUDIEN : The CLAUsal DIscovery ENgine User's Guide - version 3 .0, 1996.

[29] Y. Dimopoulos and P. Moraitis and A. Tsoukias, Argumentation Based Modeling of Decision Aiding for Autonomous Agents, 99-105, IAT, 2004.

[30] P.M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, 321-357, Essex, UK, Artif. Intell., Elsevier Science Publishers Ltd., 2, 1995.

[31] E. Fabre and A. Benveniste and C. Jard and L. Ricker and M. Smith, Distributed state reconstruction for discrete event systems, 2000.

[32] P. Finn and S. Muggleton and D. Page and A. Srinivasan, Pharmacophore Discovery Using the Inductive Logic Programming System PROGOL,.Magazine,: 241—270, Hingham, MA, USA, Mach. Learn., Kluwer Academic Publishers, 2-3, 1998.

[33] The Foundation for Intelligent Physical Agent : http://www.fipa.org,.

[34] FIPA Performative Specifications :http://www.fipa.org/specs/fipa00061/SC00061G.html.

[35] FIPA Performative Specifications : http://www.fipa.org/specs/fipa00037/SC00037J.html.

[36] FIPA Interaction Protocol Specification : http://www.fipa.org/repository/ips.php3.

[37] P.M. Frank, Enhancement of robustness in observer-based fault detection, 955-981, International Journal of Control, 1994.

[38] P. Fröhlich and I. de Almeida Mora and W. Nejdl and M. Schroeder, Diagnostic Agents for Distributed Systems, 173-186, 3-540-67027-0, London, UK, Springer-Verlag, Selected papers from the ESPRIT Project ModelAge Final Workshop on Formal Models of Agents, 1999.

[39] E. Fromont and A. Hernandez and M.O. Cordier, R. Quiniou, Kardio and Calicot: a comparison of two cardiac arrhythmia classifiers, 29-33, Artificial Intelligence in Medicine Europe, workshop Qualitative and Model-Based Reasoning in BioMedecine, 2003.

[40] E. Fromont and M.O. Cordier and R. Quiniou, Extraction de connaissances provenant de données multisources pour la caractérisation d'arythmies cardiaques, Extraction et Gestion de la Connaissance, 2004.

[41] C. Frydman and M. {Le Goc} and L. Torres and N. Giambiasi, The Diagnosis Approach Used in SACHEM,.Rapport de congrès, 63-70, Sansicario, Italy, Mars, DX'2001, 12th International Workshop on Principles of Diagnosis, 2001.

[42] W. Gerstner and W. Kistler, Spiking Neuron Models: An Introduction, 0521890799, New York, NY, USA, Cambridge University Press, 2002.

[43] J. Gertler and D. Singer, A new structural framework for parity equation-based failure detection and isolation, 381-388, Tarrytown, NY, USA, Automatica, Pergamon Press, Inc., 2, 1990.

[44] J. C. Giarratano and G. Riley, Expert Systems: Principles and Programming, 0534937446, Boston, MA, USA,PWS Publishing Co., 1994.

[45] M. Le Goc, SACHEM, a Real-Time Intelligent Diagnosis System Based on the Discrete Event Paradigm, 591-617, San Diego, CA, USA,Simulation, Society for Computer Simulation International, 11, 2004.

[46] A. Grastien, Diagnostic décentralisé et en-ligne de systèmes à événements discrets reconfigurables, Equipe DREAM - IRISA, Université de Rennes 1,2005.

[47] A. Grastien and M. O. Cordier and C. Largouët and K. Balazs and G. Lapalme, First steps towards incremental diagnosis of discrete-event systems, 170-181, Lecture notes in computer science (Lect. notes comput. sci.) ISSN 0302-9743, 2005.

[48] A. Grastien, Utilisation des chaînes d'automates pour le diagnostic décentralisé, RFIA'06, 2006.

[49] C.L. Hamblin, Fallacies, London, Methuen, 1970.

[50] D.O. Hebb, The organization of behavior, MIT Press Cambridge, MA, USA, 1988.

[51] K. Heljanko and V. Khomenko and M. Koutny, Parallelisation of the Petri Net Unfolding Algorithm, 371-385, 3-540-43419-4, London, UK, Springer-Verlag, TACAS '02: Proceedings of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, 2002.

[52] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, 457-464, 0-262-01097-6, Cambridge, MA, USA, MIT Press, 1988.

[53] J. Ignizio, Introduction to Expert Systems, 0-07-909785-5, McGraw-Hill,New York, 1991.

[54]Jess, The Java Expert System Shell Website : http://herzberg.ca.sandia.gov/jess,.

[55] A.C. Kakas and P. Moraitis, Argumentative Agent Deliberation, Roles and Context, 35-48, CLIMA III, 2002.

[56] A.C. Kakas and P. Mancarella and P.M. Dung, The Acceptability Semantics for Logic Programs, 504-519, ICLP, 1994.

[57] R. E. Kalman, A New Approach to Linear Filtering and Prediction Problems, 35-45,Transactions of the ASME Journal of Basic Engineering,1960.

[58] R.E. Kalman and R.S. Bucy, New Results in Linear Filtering and Prediction Theory, 95-107, Transactions of the ASME - Journal of Basic Engineering, 1961.

[59] N.C. Karunatillake and N.R. Jennings and I. Rahwan and T.J. Norman, Argument-based negotiation in a social context, 1331-1332,1-59593-093-0,New York, NY, USA, ACM Press, AAMAS '05 : Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, 2005.

[60] Irene Katzela and Mischa Schwartz, Schemes for fault identification in communication networks, 753-764, Piscataway, NJ, USA, IEEE/ACM Trans. Netw., IEEE Press, 6, 1995.

[61] G. Khanna and M. Yu Cheng and P. Varadharajan and M. Correia and P. Verissimo and S. Bagchi, Automated Monitor Based Diagnosis in Distributed Systems, School of Electrical and Computer Engeering, Prudue University, 2005.

[62] V. Khomenko and M. Koutny, Towards an Efficient Algorithm for Unfolding Petri Nets, 366-380, 3-540-42497-0, London, UK,Springer-Verlag, CONCUR '01: Proceedings of the 12th International Conference on Concurrency Theory, 2001.

[63] I. Kononenko, Inductive and Bayesian Learning in Medical Diagnosis, Applied Artificial Intelligence, 1993.

[64] P. Langley and H.A. Simon, Applications of machine learning and rule induction, 54-64, New York, NY, USA,Commun. ACM,ACM Press, 11, 1995.

[65] J.L. Laurière, Un Langage déclaratif : Snark, Paris : AFCET : Gauthier-Villars, 1986.

[66] A. M. Leake and C. C. Owens, Swale : A Program that Explains, Schank, R. C. (ed.), Explanation Patterns : Understanding Mechanically and Creatively. Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

[67] B. Li and M.Y. Chow and Y. Tipsuwan and J.C. Hung, Neural-network-based motor rolling bearing fault diagnosis, 1060-1069, Industrial Electronics, IEEE Transactions, 2000.

[68] Shu-Hsien Liao, Expert system methodologies and applications - a decade review from 1995 to 2004, 93-103, Expert Systems with Applications, January, 2005.

[69] I. Linnik, Method of Least Squares and Principles of the Theory of Observations, New York, Pergamon Press, 1961.

[70] J. MacKenzie, Question-begging in non-cumulative systems, Journal of Philosophical Logic, 8:117-133.

[71] E. Mangina and S.D.J. McArthur and J.R. McDonald, COMMAS (COndition Monitoring Multi-Agent System), 279-282, Autonomous Agents and Multi-Agent Systems, 3, 2001.

[72] N. Maudet and F. Evrard, A generic framework for dialogue game implementation, Proceeding of the 2nd Workshop on Formal Semantics and Pragmatics of Dialogue, Univerity of Twente, The Netherlands, 1998.

[73] S.D.J McArthur and E. Davidson, Multi-agent systems for diagnostic and condition monitoring applications, Power Engineering Society General Meeting, 2004, IEEE.

[74] W.S. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity, 15-27,0-262-01097-6, Cambridge, MA, USA, MIT Press, 1988.

[75] P. Moraitis, Un modèle de raisonnement pour agents autonomes fondé sur l'argumentation, In Proc. Journées Nationales de Modèles de Raisonnement, 2003.

[76] M. Morge and J. Dumont, Cadre d'argumentation bayésien, Actes des JFRB'06, 2006.

[77] M. Morge, Système dialectique multi-agents pour l'aide à la concertation, 2005.

[78] S. Muggleton, Inductive logic programming: derivations, successes and shortcomings, 5-11, New York, NY, USA, SIGART Bull, ACM Press, 1, 1994.

[79] S. Muggleton and C. Feng, Efficient induction of logic programs, 368-381, Ohmsma, Tokyo, Japan, Proceedings of the 1st Conference on Algorithmic Learning Theory, 1990.

[80] S. Muggleton, Inductive logic programming: derivations, successes and shortcomings, 5-11, New York, NY, USA, SIGART Bull, ACM Press, 1, 1994.

[81] S. Muggleton, Inverse Entailment and Progol, 245-286, New Generation Computing, Special issue on Inductive Logic Programming, Ohmsha, 3-4, 1995.

[82] P. Nii, The blackboard model of problem solving, 38-53, Menlo Park, CA, USA, AI Mag., American Association for Artificial Intelligence, 2, 1986.

[83] INSERM and Institut Roche de l'Obésité and SOFRES, ObEpi2000 : Le surpoids et l'obésité en France. Enquête épidémiologique réalisée dans un échantillon représentatif de la population française, adulte et enfant.

[84] Judea Pearl, Bayesian networks, 149-153, 0-262-51102-9, Cambridge, MA, USA, The handbook of brain theory and neural networks, MIT Press, 1998.

[85] D. Pelleg and A.W. Moore, X-means: Extending K-means with Efficient Estimation of the Number of Clusters, 727-734, Seventeenth International Conference on Machine Learning, 2000.

[86] Y. Pencolé, Diagnostic décentralisé de systèmes à événements discrets : application aux réseaux de télécommunications, Equipe DREAM - IRISA, Université de Rennes 1, 2002.

[87] J. Peterson, Petri Net Theory and the Modeling of Systems, N.J.: Prentice-Hall, Englewood Cliffs, 1981.

[88] D. Poole, On the Comparison of Theories: Preferring the Most Specific Explanation, 144-147, IJCAI, 1985.

[89] B. Porter and E. R. Bareiss, PROTOS : an experiment in knowledge acquisition for heuristic classification tasks, University of Texas at Austin,Artificial Intelligence Laboratory, 1986.

[90] H. Prakken and G. Vreeswijk, Logics for defeasible argumentation, 219-238, Handbook of Philosophical Logic, Vol. 4, Kluwer Academic Publishers, Dordrecht, Netherlands, second Edition, 2002.

[91] H. Prakken and G. Sartor, Argument-Based Extended Logic Programming with Defeasible Priorities, Journal of Applied Non-Classical Logics, 1, 1997.

[92] J. R. Quinlan, Induction of Decision Trees, 81-106, Hingham, MA, USA, Mach. Learn., Kluwer Academic Publishers, 1, 2003.

[93] J. Ross Quinlan, C4.5: programs for machine learning, 1-55860-238-0, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 1993.

[94] J. Ross Quinlan and R. Mike Cameron-Jones, Induction of Logic Programs: FOIL and Related Systems, 287-312, New Generation Computing, 3&4, 1995.

[95] I. Rahwan and S.D. Ramchurn and N.R. Jennings and P. Mcburney and S. Parsons and L. Sonenberg, Argumentation-based negotiation, 343-375, New York, NY, USA, Knowl. Eng. Rev., Cambridge University Press, 4, 2003.

[96] C. Reed, Dialogue Frames in Agent Communication, 246, 0-8186-8500-X,Washington, DC, USA, IEEE Computer Society, ICMAS '98: Proceedings of the 3rd International Conference on Multi Agent Systems, 1998.

[97] J.M. Renders and A. Goosens and F. de Viron and M. De Vlaminc, A prototype neural network to perform early warning in nuclear power plant, 139-151, Fuzzy Sets and Systems, 1995.

[98] L.A.M. Riascos and M.G. Simoes and P.E. Miyagi, A Bayesian network fault diagnostic system for proton exchange membrane fuel cells, Journal of Power Sources, Elsevier, 2007.

[99] N. Roos and A. Teije and A. Bos and C. Witteveen, An analysis of multi-agent diagnosis, 2002.

[100] N. Roos and A. ten Teije and C. Witteveen, A protocol for multi-agent diagnosis with spatially distributed knowledge, 655-661, 1-58113-683-8, New York, NY, USA, ACM Press, AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems, 2003.

[101] F. Rosenblatt, The perception: a probabilistic model for information storage and organization in the brain, 89-114, 0-262-01097-6, Cambridge, MA, USA, MIT Press, 1988.

[102] L. Rozé and M.O. Cordier, Diagnosing Discrete-Event Systems: Extending the Diagnoser Approach to Deal with Telecommunication Networks, 43-81, Hingham, MA, USA, Discrete Event Dynamic Systems, Kluwer Academic Publishers, 1, 2002.

[103] S.J. Russell and P. Norvig, Artificial intelligence: a modern approach, 0-13-103805-2, Upper Saddle River, NJ, USA, Prentice-Hall, Inc., 1995.

[104] F. Sahi and M. C. Yavuz and Z. Arnavut and O. Uluyol, Fault diagnosis for airplane engines using Bayesian networks and distributed particle swarm optimization, Parallel Computing, Elsevier, 2007.

[105] M. Sampath and R. Sengupta and S. Lafortune and K. Sinnamohideen and D. Teneketzis, Diagnosability of discrete-event systems, 1555 – 1575, Automatic Control, IEEE Transactions, September, 1995.

[106] R. Schank, Dynamic Memory: A Theory of Learning in Computers and People (New York: Cambridge University Press, 1982), New York: Cambridge University Press, 1982.

[107] R. Schank, Explanation Patterns: Understanding Mechanically and Creatively, Erlbaum, 1986.

[108] M. Schroeder and G. Wagner, Vivid agents: theory, architecture, and applications, 2000.

[109] R. Schweimeier and M. Schroeder, Notions of Attack and Justified Arguments for Extended Logic Programs, 536-540, ECAI, 2002.

[110] M. Schroeder and G. Wagner, Distributed diagnosis by vivid agents, 268-275, 0-89791-877-0, New York, NY, USA, ACM Press, AGENTS '97: Proceedings of the first international conference on Autonomous agents, 1997.

[111] E.H. Shortliffe, Mycin: a rule-based computer program for advising physicians regarding antimicrobial therapy selection, 1975.

[112] E.H. Shortliffe, Computer-based medical consultations, MYCIN, Elsevier, New York, 1976.

[113] C. Sierra and N.R. Jennings and P. Noriega and S. Parsons, A Framework for Argumentation-Based Negotiation, 177-192, 3-540-64162-9, London, UK, Springer-Verlag, ATAL '97: Proceedings of the 4th International Workshop on Intelligent Agents IV, Agent Theories, Architectures, and Languages, 1998.

[114] G.R. Simari and R.P. Loui, A Mathematical Treatment of Defeasible Reasoning and its Implementation, 125-157, Artif. Intell., 2-3, 1992.

[115] L. Spalzzi, A Survey on Case-Based Planning, 3-36, Norwell, MA, USA, Artif. Intell. Rev., Kluwer Academic Publishers, 1, 2001.

[116] R. Srikant and R. Agrawal, Mining Generalized Association Rules, 407-419, 1-55860-379-4, Umeshwar Dayal and Peter M. D. Gray and Shojiro Nishio,Morgan Kaufmann,VLDB'95, Proceedings of 21th International Conference on Very Large Data Bases, September 11-15, 1995, Zurich, Switzerland, 1995.

[117] R. Srikant and R. Agrawal, Mining quantitative association rules in large relational tables, 1-12, 0-89791-794-4, New York, NY, USA, ACM Press, SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data, 1996.

[118] Gerard Vreeswijk, Argumentation in Bayesian Belief Networks, 111-129, ArgMAS, 2004.

[119] G. Wagner, Vivid Agents -- How They Deliberate, How They React, How They Are Verified.

[120] D.N. Walton and E.C. Krabbe, Commitment ini Dialogue: Basic Concepts of Interpersonal Reasoning, State University of New York Press, Albany, NY, 1995.

[121] F. Wang and R. Quiniou and G. Garrault and M.O. Cordier, Learning Structural Knowledge from the ECG, 288-294, 3-540-42734-1, London, UK, Springer-Verlag, ISMDA '01: Proceedings of the Second International Symposium on Medical Data Analysis, 2001.

[122] , Multiagent systems: a modern approach to distributed artificial intelligence, 0-262-23203-0, Cambridge, MA, USA, Gerhard Weiss, MIT Press, 1999.

[123] M. Woolridge and M. J. Wooldridge, Introduction to Multiagent Systems,.Livre,: , 047149691X,New York, NY, USA,John Wiley \& Sons, Inc.,2001.

[124] P. Xu and S. Xu and H. Yin, Application of self-organizing competitive neural network in fault diagnosis of suck rod pumping system, Journal of Petroleum Science and Engineering, 2007.

[125] S. Zad and R. Kwong and W. Wonham, Fault diagnosis in discrete-event systems: Framework and model reduction, 1998.

École Nationale Supérieure des Mines
de Saint-Étienne

N° d'ordre : ……

Julien DUMONT

Multi-Agent System for Multi-Disciplinary Diagnostic

Speciality : Computer Science

Keywords : Multi-Agent System, Bayesian Network, Argumentation

Abstract :

Sharing opinions among different participants is a useful and common way to build a constructive argumentation in order to solve complex problems that require the confrontation of different discipline areas. In such settings, experts build different arguments in relation to their own discipline area, then share and confront them to the other experts' opinions. In this report we present an argumentative framework ANDi based on a multi-agent approach and Bayesian networks. In this framework, the agents support the elaboration of a global diagnostic from local ones. Local diagnostics are resulting of argumentations between group of experts from the same discipline area. We illustrate the use of this argumentation framework on the domain of fault diagnosis