Raisonnement sous incertitude et en présence de préférences : Application à la détection d'intrusions et à la corrélation d'alertes

THÈSE

soutenue publiquement le 04 Décembre 2008

pour l'obtention du

Doctorat de l'Université d'Artois

Spécialité Informatique

par

Karima SEDKI

Composition du jury

Rapporteurs : Philippe LERAY, Professeur des Universités, Polytech'Nantes
Stéphane LOISEAU, Professeur des Universités, Université d'Angers

Examinateurs : Salem BENFERHAT, Professeur des Universités, Université d'Artois *(Directeur de thèse)*
Gilles GONCALVES, Professeur des Universités, Université d'Artois
Benjamin MORIN, Ingénieur de recherche, Direction centrale de la sécurité des systèmes d'information
Lakhdar SAÏS, Professeur des Universités, Université d'Artois

Centre de Recherche en Informatique de Lens – CNRS UMR 8188
Université d'Artois, rue Jean Souvraz, S.P. 18 F-62307, Lens Cedex France
Secrétariat : Tél.: +33 (0)3 21 79 17 23 – Fax : +33 (0)3 21 79 17 70
http://www.cril.fr

Résumé

Cette thèse a pour objectif le traitement de deux problèmes importants : représentation des préférences avec application à la corrélation d'alertes et raisonnement sous incertitude avec application à la détection d'intrusions. Dans un premier temps, nous nous sommes intéressés à la représentation des préférences dans un cadre logique. Nos travaux se sont focalisés sur des extensions de la logique du choix qualitatif ($\mathcal{QCL}$) pour représenter des préférences complexes. Notre objectif est de proposer de nouvelles logiques afin de remédier aux limites de $\mathcal{QCL}$ et de pouvoir représenter différents types de préférences telles que les préférences prioritaires et les préférences positives. Par ailleurs, cette thèse vise aussi à résoudre le problème de la corrélation d'alertes qui consiste à réduire les grandes quantités d'alertes générées par les systèmes de détection d'intrusions (IDSs) en proposant une approche qui permet d'intégrer les connaissances et les préférences d'un administrateur au sujet des alertes qu'il préfère analyser ou ignorer. L'idée consiste à coder les connaissances et les préférences exprimées dans le cadre de l'une des logiques que nous avons proposées et de présenter à l'administrateur uniquement les alertes préférées. Dans un second temps, nous nous sommes intéressés au problème de la détection d'intrusions qui relève de la sécurité des systèmes d'informations. Notre objectif est de détecter des attaques réseaux présentées dans des connexions finies ou dans des connexions non finies. Ainsi, nous avons proposé de modéliser ce problème par un modèle graphique probabiliste. Pour cela, nous avons défini un jeu d'attributs permettant de décrire les données réseaux et de distinguer un trafic normal d'un trafic intrusif. Nous avons développé un outil de formatage pour extraire les attributs définis et transformer les données réseaux brutes en données formatées. Notre outil permet de structurer les données réseaux en connexions finies (complètes) et connexions non finies (incomplètes). Ces connexions sont utilisées pour la détection en différé et/ou en temps réel des attaques.
Mots clés : représentation des préférences, corrélation d'alertes, détection d'intrusions, réseaux bayésiens, classification.

Abstract

This thesis aims to study two important problems : representation of preferences with application to alert correlation and reasoning under uncertainty with application to intrusion detection problem. We focused on extensions of Qualitative Choice Logic ($\mathcal{QCL}$) to represent complex preferences. Our aim is to propose new logics to address $\mathcal{QCL}$ limitations and to represent different kinds of preferences such as prioritized and positive preferences. Moreover, this thesis aims to solve the problem of alert correlation that concerns reducing the large amounts of generated alerts by intrusion detection systems (IDSs). We propose an approach that incorporates administrator's knowledge and preferences about alerts that he prefers to analyze or ignore. The idea is to encode administrator's knowledge and preferences in the context of one of our developed logics and present to the administrator only preferred alerts. On the other hand, we are interested in the problem of intrusion detection that is a serious problem in computer security. Our objective is to detect network attacks before or after connection's completion. Thus, we propose to represent this problem by a probabilistic graphical model used for classification purposes. To do this, we defined a set of attributes that describe network data and distinguish between normal and intrusive traffic. We have developed a preprocessing tool to extract the defined attributes and transform raw data to formatted one. Our tool can structure network data into finished or not finished connections that can be used for on-line or/and off-line attacks detection.
Keywords : representation of preferences, alert correlation, intrusion detection, bayesian networks, classification.

Remerciements

Je tiens à remercier tout d'abord Salem BENFERHAT pour m'avoir encadrée et encouragée tout au long de ces trois années de thèse. Je le remercie aussi pour ses conseils et orientations pertinentes, et surtout pour m'avoir permis de découvrir ce domaine de recherche que je trouve très passionnant.

Je remercie également Philippe LERAY et Stéphane LOISEAUX d'avoir accepté d'être rapporteurs de cette thèse. Je remercie aussi Gilles GONCALVES, Benjamin MORIN et Lakhdar SAÏS d'avoir accepté de faire partie du jury de cette thèse.

Mes grands remerciements vont aussi à tous les membres du CRIL (faculté et IUT) pour leur sympathie et gentillesse. Merci aussi aux doctorants avec lesquels j'ai passé de très bons moments durant ces trois années.

Je remercie également le personnel des départements informatique, SRC et GEA dans lesquels j'effectue mes enseignements.

Je profite aussi de cette occasion qui m'est offerte pour remércier les équipes du projet DADDi et PLACID.

Une petite pensée à ma famille, qui sans elle rien n'aurait abouti. Merci à mes parents pour leurs encouragements, leurs suivis et pour tout ce qu'ils m'ont donné pour mener à bien mes études. Un grand merci également à mes soeurs et mes frères, sans oublier le reste de ma famille.

Enfin, je remercie toutes les personnes que je n'ai, malheureusement, pas citées mais je veux leur dire combien leur présence est importante pour moi.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

L'intelligence artificielle est une discipline qui a pour objectif de modéliser les raisonnements humains. Parmi les problèmes traités, on trouve la représentation des connaissances, la modélisation des préférences, l'apprentissage, l'aide à la décision, la planification, etc. Dans une grande partie de cette thèse, nous nous intéressons au problème de représentation et modélisation des préférences. Ce problème est important du fait que l'on est souvent dans des situations de choix parmi un ensemble de propositions ou d'objets à comparer ou à évaluer.

Pour représenter les préférences, deux principales catégories d'approches sont utilisées : les approches qualitatives ou logiques et les approches quantitatives ou numériques. La représentation logique des préférences a été largement explorée dans de nombreux travaux ([Dubois *et al.* 2002], [Boutilier 1994], [Boutilier *et al.* 1999], [Lang 1996], [Lang *et al.* 2002], [Brewka 2002]) car elle offre un cadre formel bien défini et une simplicité d'utilisation. Toutefois, certaines logiques se sont montrées insuffisantes. Elles ne permettent de représenter que des préférences d'une forme et d'une structure bien particulière, alors que différents types de préférences existent (conditionnelles, prioritaires, uni-modales, etc.). A l'exception de la logique du choix qualitatif ($\mathcal{QCL}$) [Brewka *et al.* 2004], les approches développées ne permettent pas de raisonner sur les préférences de la forme *"Si A est préféré à B alors C est préféré à D"*. Or, même si cette logique donne la possibilité d'exprimer ce type de préférences, sa relation d'inférence n'est pas satisfaisante. Notre travail consiste à développer d'autres logiques permettant de remédier aux problèmes de la logique $\mathcal{QCL}$, d'exploiter ses points positifs, de représenter d'autres types de préférences et de les appliquer au problème de la corrélation d'alertes par exemple.

Les techniques d'intelligence artificielle sont de plus en plus utilisées pour des problèmes très sensibles et préoccupants telle que la sécurité des systèmes d'informations. En effet, la quasi-totalité des secteurs professionnels tels que les banques, les hôpitaux, les universités, sont tous informatisés et disposent de technologies très avancées. La connectivité à l'Internet et la possibilité de travailler en réseau sont parmi les technologies qui attirent de plus en plus d'utilisateurs. Or, une partie de ces utilisateurs, qu'ils soient internes ou externes, n'ont pas forcément de bonnes intentions vis-à-vis de ces systèmes. Ils peuvent perturber leur fonctionnement en effectuant différentes opérations comme la consultation et modification de données sensibles.
La sécurité de ces systèmes est ainsi devenue un problème sensible et un enjeu majeur. C'est pourquoi, plusieurs mécanismes et moyens matériels et logiciels ont été développés, afin d'assurer la protection de ces systèmes des utilisations malveillantes. Parmi ces outils, on trouve les parefeux qui permettent de contrôler les flux de données échangés, les mécanismes d'authentification qui permettent aux utilisateurs qui tentent d'accéder aux systèmes de prouver leur identité, les mécanismes de contrôles d'accès qui accordent les privilèges nécessaires aux utilisateurs, des analyseurs permettant d'analyser le trafic circulant réseau, etc.

Notre travail consiste à appliquer des techniques d'intelligence artificielle à la sécurité des systèmes d'informations, notamment à la détection d'intrusions.

1 Contexte de la thèse et problématique

Cette thèse s'inscrit dans le cadre de l'intelligence artificielle. Nous nous intéressons à traiter deux problèmes importants : représentation des préférences et application des techniques d'intelligence artificielle (modèles graphiques et logique de représentation des préférences) au problème de la détection d'intrusions.

Représentation des préférences : Concernant la représentation des préférences, nous nous intéressons à une approche logique appelée "*Logique du Choix Qualitatif ($\mathcal{QCL}$)*". La logique $\mathcal{QCL}$ [Brewka *et al.* 2004] est une nouvelle logique de représentation des préférences qui ajoute à la logique propositionnelle classique un nouveau connecteur, nommé *disjonction ordonnée*, symbolisée par $\vec{\times}$, qui permet d'ordonner les différentes alternatives sur lesquelles les agents définissent leurs préférences. La formule "$A \vec{\times} B$" signifie "A si c'est possible, mais si A est impossible alors au moins B". Cette logique est intéressante car elle donne la possibilité d'exprimer des préférences simples de la forme *"A est préféré à B"* ou complexes de la forme (*"Si A est préféré à B alors C est préféré à D"*, *"(A préféré à B) ET (C préféré à D)"*, etc.). Les préférences simples sont représentées par des *formules de choix de base (BCF)* et les préférences complexes sont représentées par des *formules de choix générales (GCF)*. La sémantique d'une formule $\mathcal{QCL}$ (BCF ou GCF) est basée sur son degré de satisfaction étant donnée une interprétation. Les alternatives qui satisfont une formule donnée avec le plus petit degré sont considérées les meilleures.

Une étude détaillée de cette logique nous a amenés à nous focaliser sur quelques points essentiels qui mènent à restreindre son champ d'application. Nous discuterons plus précisément des cas du raisonnement conditionnel sur les préférences et de la négation d'un ensemble de préférences. La relation d'inférence $\mathcal{QCL}$ n'est pas totalement satisfaisante [Benferhat *et al.* 2006, Benferhat & Sedki 2008b]. En effet, les préférences exprimées par des règles conditionnelles de la forme *"Si A est préféré à B alors C est préféré à D"* sont équivalentes aux préférences de la forme *"Si A ou B alors C est préféré à D"*. Ce problème est dû à la définition de la négation, qui ignore la notion de préférence. En effet, la préférence représentée par la formule de forme *"¬(A $\vec{\times}$ B)"* est équivalente à la formule propositionnelle *"¬(A ∨ B)"*, où le symbole de préférence est remplacé par la disjonction propositionnelle. Cela pose une vraie limite de la logique $\mathcal{QCL}$, alors que la négation a une importance capitale pour exprimer les préférences par des rejets (présence de la négation dans ces préférences) et des règles conditionnelles.

Détection d'intrusions : la détection d'intrusions est un problème préoccupant qui vise à identifier les activités malveillantes afin de détecter des attaques. Elle est apparue au début des années 80, suite aux travaux fondateurs dans le domaine [Anderson 1980, Denning 1987]. Comme il n'est pas envisageable de détecter les intrusions manuellement vu le volume important des données à analyser, des systèmes de détection d'intrusions (IDSs) ont été développés pour détecter automatiquement les activités malveillantes qui visent le système d'informations surveillé. Actuellement, deux principales approches de détection sont utilisées par les IDSs : *"approche comportementale"* et *"approche par signatures ou par scénarios"*. L'approche comportementale consiste à définir un profil représentant les différentes activités normales des utilisateurs, services et applications au sein du système. Toute déviation par rapport au comportement

normal sera interprétée comme une éventuelle intrusion. Comme exemples de paramètres pris en considération pour modéliser les activités normales, on peut trouver le temps d'utilisation des ressources, répartition statistique des protocoles et applications utilisés, volume des données échangées, etc. L'approche par signatures s'appuie sur la connaissance des stratégies des attaquants pour déduire des scénarios typiques. Elle utilise des signatures d'attaques (spécifications propres de l'attaque : une chaîne alphanumérique, accès à un fichier système, etc.). Ces techniques utilisent différentes méthodes : statistiques [Porras & Neumann 1997], systèmes experts [Habra *et al.* 1992, Valdes & Skinner 2000], fouille de données [Lee *et al.* 2000], algorithmes génétiques [Mé 1994], etc.

Ces approches offrent plusieurs avantages comme la capacité de détecter des attaques connues. Cependant, plusieurs problèmes demeurent sans solutions satisfaisantes. Les principaux problèmes auxquels nous nous intéressons sont les suivants :

- Toutes les approches analysant les connexions ne détectent pas les intrusions en temps réel à proprement parler car on attend la fin d'une connexion afin extraire ses attributs pour l'analyser. Cela peut être très dommageable car le temps que la connexion se termine, l'attaquant aura accompli son forfait.
- Étant donné un trafic réseau brut, l'analyse n'est pas une tâche facile, vue la nature des données qui est brute et la quantité de données qui est très volumineuse.
- Le volume de fausses alertes générées est très important, notamment dans le cas d'une approche comportementale lorsque, par exemple, des utilisateurs autorisés changent leurs comportements habituels au sein du système.

Afin de pouvoir détecter des activités malveillantes dans un réseau, il est important de définir les informations nécessaires à analyser. Or, cette tâche n'est pas facile, car plusieurs éléments peuvent intervenir dans la réalisation des attaques : types des protocoles et services, moments et durées des attaques, nature des ressources et applications exploitées, types et nombre d'actions exécutées, etc. En fait, la difficulté majeure d'un problème de détection d'intrusions est principalement liée à la complexité des données à analyser, car les données réseaux sont de nature brute, souvent hétérogènes (différentes sources) et disponibles en grandes quantités (plusieurs giga-octets). Comme les attaques sont réalisées au niveau des paquets, il n'est donc pas possible de pouvoir les analyser telles qu'elles sont. Ainsi, nous considérons que le problème de la détection d'intrusions est, avant tout, un problème de description des données réseaux qui nécessitent deux éléments importants : la définition et l'extraction des informations (caractéristiques d'activités normales et attaques).

Un autre problème important de la détection d'intrusions que nous avons déjà souligné plus haut, concerne le nombre important d'alertes que les IDSs produisent. La majorité de ces alertes ne correspondent pas réellement à des attaques (fausses alertes, alertes redondantes, etc.). Ainsi, l'administrateur qui a pour tâche d'analyser et de prendre des décisions nécessaires, se retrouve rapidement débordé, et laisse certainement passer des alertes suspectes sans les analyser. Ce problème relève de la corrélation d'alertes qui est une branche de la détection d'intrusions. D'une manière générale, *la corrélation d'alertes est définie comme l'interprétation conceptuelle de plusieurs alertes afin de leur attribuer une meilleure sémantique d'une part et de réduire le volume important d'alertes d'autre part* [Jakobson & Weissman 1993]. Plusieurs approches de corrélation d'alertes [Eckmann *et al.* 2002, Valdes & Skinner 2001, Cuppens & Miège 2002, Staniford *et al.* 2002, Morin *et al.* 2002, Ning *et al.* 2002, Julisch 2003] ont été développées dans

le but de remédier à ce problème. Cependant, malgré l'efficacité de la majorité de ces approches pour éliminer les informations redondantes ou détecter des attaques coordonnées et complexes ; les systèmes de détection d'intrusions continuent à produire des quantités volumineuses d'alertes. Dans cette thèse, nous nous intéressons également à ce problème.

2 Objectifs et approche envisagée

Compte tenu des limites posées par la logique du choix qualitatif $\mathcal{QCL}$ pour la représentation des préférences, nous nous intéressons à :

- Cerner les limites de cette logique.
- Définir un cadre qui nous permettra de remédier aux limites de cette logique.
- Modifier le langage de la logique $\mathcal{QCL}$, afin de pouvoir représenter d'autres types de préférences telles que les prioritaires et positives.
- Étendre le langage de $\mathcal{QCL}$ dans le cadre de la logique du premier ordre afin de pouvoir modéliser le problème de la corrélation d'alertes dans le nouveau cadre.

D'un côté, l'approche que nous envisageons pour remédier à ce problème consiste principalement à définir autrement la négation dans les préférences. De plus, comme nous nous intéressons à la représentation d'autres types de préférences, nous envisageons également d'apporter d'autres modifications à la logique QCL au niveau de la conjonction et de la disjonction des préférences.

Pour le problème de la détection d'intrusions, nous nous intéressons à définir et à extraire les informations essentielles afin de pouvoir repérer facilement des paquets suspects et de détecter des attaques. Comme nous l'avons déjà souligné, les données réseaux sont brutes ; pour pouvoir les analyser, il est nécessaire de les transformer dans des formats exploitables par des techniques de détection. Nos principaux objectifs sont les suivants :

- Réduction de la quantité des données à analyser.
- Définir un jeu d'attributs pertinents pour pouvoir décrire les données réseaux et distinguer les activités normales et anormales.
- Structurer les données réseaux qui sont de nature brute pour construire des connexions finies et des connexions non finies que nous utiliserons pour la détection en temps réel.

Nous nous intéressons particulièrement à l'utilisation des modèles graphiques probabilistes (également appelés réseaux bayésiens) [Jensen 2001, Pearl 1988, Naïm *et al.* 2007] à des fins de classification [Friedman *et al.* 1997]. Les modèles graphiques sont des formalismes efficaces pour la représentation et raisonnement avec des informations incertaines et complexes. Ils sont utilisés dans de nombreux domaines : en représentation des préférences [Boutilier *et al.* 2004], dans les problèmes de décisions [Qi & Poole 1995], dans les problèmes de satisfaction de contraintes [Montanari 1974] et dans beaucoup d'autres problèmes de raisonnement probabiliste et possibiliste [Jaeger 2004, Borgelt *et al.* 2002, Benferhat & Smaoui 2007], etc. En détection d'intrusions, l'utilisation des ces formalismes s'est avérée intéressante grâce à leurs aspects algorithmiques (en apprentissage automatique, inférence, classification, etc.) et modulaires.

Pour modéliser le problème de détection d'intrusions à l'aide d'un modèle graphique probabiliste, il est nécessaire, dans un premier temps, de représenter les données du problème dans un format exploitable par ces modèles. C'est pourquoi, nous proposons de définir un jeu d'attributs

décrivant les données réseaux, de développer un outil de formatage afin de formater les données brutes, extraire les attributs définis et construire des connexions. Une fois les données formatées, c'est-à-dire qu'elles sont représentées sous forme de connexions décrites par une liste d'attributs, nous appliquons alors une technique d'apprentissage automatique/classification pour construire le classifieur qui va être utilisé pour déterminer la classe d'une nouvelle connexion (normale ou attaque). Les techniques de classification et d'apprentissage automatique permettent d'élaborer sur les données d'apprentissage préalablement étiquetées, des modèles de classification pouvant être généralisés sur l'ensemble des données du domaine. Nous nous intéressons en particulier à la détection d'intrusions avec des connexions non finies pour analyser le trafic réseau en temps réel véritablement.

Concernant le problème du taux d'alertes générées par les systèmes de détection d'intrusions, nos objectifs sont :

- Proposer une nouvelle approche de corrélation d'alertes permettant de prendre en considération les connaissances et les préférences d'un administrateur réseau.
- Modéliser les connaissances et les préférences de l'administrateur dans un cadre logique.
- Ordonner les alertes selon les connaissances et les préférences de l'administrateur réseau.

3 Principales contributions

Les principales contributions de nos travaux de thèse sont résumés dans les points suivants :

1. *Révisions et modifications de la logique $\mathcal{QCL}$:* concernant la révision de la logique $\mathcal{QCL}$, nous avons proposé trois alternatives permettant de remédier aux limites de $\mathcal{QCL}$. Ces alternatives concernent trois logiques différentes : $\mathcal{MQCL}$ (*Logique du Choix Qualitatif Minimale*), $\mathcal{PQCL}$ (*Logique du Choix Qualitatif Prioritaire*) et $\mathcal{QCL}+$ (*Logique du Choix Qualitatif Positive*). Les trois logiques se basent sur le même langage que celui de la logique $\mathcal{QCL}$, mais chacune utilise des règles d'inférence différentes. Ce qui caractérise chaque logique est une nouvelle définition de la négation des préférences qui permet de surmonter les limites de $\mathcal{QCL}$. La logique $\mathcal{MQCL}$ permet de maintenir la conjonction et la disjonction de base, mais nous avons proposé une nouvelle définition de la négation. Cette nouvelle logique peut être suffisante pour certains problèmes comme la corrélation d'alertes. Elle n'est pas adaptée à représenter d'autres types de préférences comme les préférences prioritaires et positives par exemple. C'est pourquoi, nous avons apporté d'autres modifications à la logique $\mathcal{QCL}$ au niveau de la conjonction et de la disjonction des préférences, d'où les deux logiques $\mathcal{PQCL}$ et $\mathcal{QCL}+$ [Benferhat & Sedki 2008b]. La logique $\mathcal{PQCL}$ permet de représenter les préférences des agents ayant différents niveaux d'importance, c'est ce que l'on appelle *préférences prioritaires*. La logique $\mathcal{QCL}+$ est particulièrement adaptée à la représentation des *préférences positives* [Dubois *et al.* 2004], c'est-à-dire aux préférences qui permettent de représenter ce qui est souhaitable par l'agent et n'excluent jamais de solutions. Nos logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$ offrent aussi la possibilité de normalisation des préférences à l'aide de fonctions de normalisation. La normalisation des préférences est très importante du fait qu'elle permet de transformer des préférences dans d'autres formats [Coste-Marquis & Marquis 2004].

2. *Définition d'attributs et formatage du trafic réseau brut :* pour modéliser les données réseaux par un modèle graphique à des fins de classification, nous avons défini plusieurs types d'attributs : certains sont communs à tous les événements (normaux ou anormaux) comme l'adresse IP source, numéro de port, service, etc. D'autres sont de haut niveau permettant de décrire les actions des attaques, comme le nombre de tentatives d'accès en mode root par exemple, etc. Le jeu d'attributs que nous avons défini contient en tout 44 attributs. Pour extraire les attributs définis, nous avons développé un outil de formatage [Benferhat *et al.* 2007]. Cet outil permet de transformer les données brutes en données formatées, et de construire des connexions décrites par les attributs définis. Deux types de formatage sont proposés : formatage hors ligne et formatage en temps réel. Le formatage hors ligne consiste à construire des connexions finies ou non finies à partir d'un trafic réseau sauvegardé dans des fichiers. Le formatage en temps réel consiste à construire des connexions finies ou non finies (incomplètes) à partir d'un trafic réseau en cours de capture.

3. *Détection d'intrusions :* nos contributions concernent deux types de détection d'intrusions : détection avec des connexions finies et détection avec des connexions non finies. La détection avec des connexions finies est intéressante du fait que ce type de connexions permet de retracer les activités complètes d'une attaque. Quant à la détection avec des connexions non finies, notre objectif est de pouvoir identifier en temps réel un trafic malveillant sur le réseau, c'est-à-dire détecter le plutôt possible les attaques, ce qui exige l'analyse de connexions non finies.

4. *Nouvelle approche de corrélation d'alertes :* notre contribution se résume principalement à la proposition d'une nouvelle approche de corrélation d'alertes [Benferhat & Sedki 2008a]. Notre modèle permet de faciliter la tâche de l'administrateur réseau, en réduisant le grand nombre d'alertes générées. L'idée principale consiste à prendre en considération les informations contenues dans les alertes, les préférences de l'administrateur réseau ainsi que ses connaissances sur le système d'informations surveillé, sur le fonctionnement de l'IDS utilisé, ou toute autre information qui peut servir à faciliter la tâche d'analyse. Ainsi, nous nous sommes intéressés à l'utilisation de la logique de représentation de préférences que nous avons développée $\mathcal{MQCL}$ afin de pouvoir modéliser les éléments pris en considération. Dans un premier temps, nous avons proposé de généraliser le langage $\mathcal{MQCL}$ en un fragment de la logique du premier ordre qui est un langage riche permettant d'exprimer des informations générales. Dans ce nouveau langage, les préférences sont représentées par des formules de premier ordre universellement quantifiées : des formules de choix de base universellement quantifiées pour des préférences simples et formules de choix générales universellement quantifiées pour des préférences plus générales et complexes. Les connaissances ou contraintes d'intégrités sont représentées par des formules propositionnelles universellement quantifiées.

Concernant les résultats de nos différentes contributions, nous avons utilisé deux bases de données différentes. La base de données DARPA'99 [Darpa 1999] contenant du trafic réseau normal et plusieurs attaques et une autre base concernant du trafic réel et récent collecté dans le cadre du projet DADDi (Dependable Anomaly Detetion with Diagnosis).

4 Organisation de la thèse

Cette thèse est composée de deux parties englobant huit chapitres.
La première partie traite le problème de modélisation et représentation des préférences. Elle est composée de quatre chapitres :

Le chapitre 1 propose un état de l'art sur la représentation des préférences quantitatives et qualitatives.

Le chapitre 2 présente les concepts essentiels de la logique du choix qualitatif ($\mathcal{QCL}$), la sémantique, la syntaxe et plusieurs définitions telles que la notion du degré de satisfaction d'une formule $\mathcal{QCL}$, l'optionalité, la notion d'équivalence, les modèles préférés, etc.

Le chapitre 3 présente nos principales contributions concernant la représentation des préférences. Ce chapitre commence par cerner les limites de la logique $\mathcal{QCL}$. Ensuite, nous présentons les solutions que nous avons proposées pour remédier aux différents problèmes posés par ces limites. Nous présentons en détail les révisions et les modifications que nous avons apportées à la logique $\mathcal{QCL}$ qui concernent essentiellement les définitions de la négation, conjonction et disjonction des préférences. Ces modifications ont donné naissance aux trois nouvelles logiques : logique $\mathcal{MQCL}$ qui est une extension simple de $\mathcal{QCL}$, logique des préférences prioritaires ($\mathcal{PQCL}$) et logique des préférences positives ($\mathcal{QCL}+$). Dans ce chapitre, nous présentons les limites et les révisions de $\mathcal{QCL}$ ainsi que la première alternative $\mathcal{MQCL}$.

Le chapitre 4 aborde la suite de nos contributions proposées dans le chapitre 3. Nous présentons la logique des préférences prioritaires ($\mathcal{PQCL}$) et la logique des préférences positives ($\mathcal{QCL}+$). Nous étudions également les principales propriétés de nos logiques et leurs relations avec la logique possibiliste.

La deuxième partie présente nos contributions concernant l'application des modèles graphiques probabilistes à la détection d'intrusions et logique des préférences à la corrélation d'alertes. Elle est composée de quatre chapitres : les chapitres 5 et 6 proposent un état de l'art, les chapitres 7 et 8 présentent nos contributions concernant les problèmes que nous avons cités en détection d'intrusions et corrélation d'alertes.

Le chapitre 5 présente un état de l'art sur la détection d'intrusions et la corrélation d'alertes. Concernant la détection d'intrusions, nous abordons l'intérêt de la détection d'intrusions, les systèmes de détection d'intrusions ainsi que les deux principales approches de détection utilisées (approche comportementale et approche par scénarios). Pour la corrélation d'alertes, nous présentons quelques exemples d'approches développées et leurs objectifs.

Le chapitre 6 propose un état de l'art sur les modèles graphiques en général et les réseaux bayésiens en particulier. Nous décrivons brièvement l'apprentissage, l'inférence et la classification dans les réseaux bayésiens. Nous présentons également l'utilisation des réseaux bayésiens en détection d'intrusions.

Le chapitre 7 traite le la définition et l'extraction des attributs décrivant les connexions que nous utiliserons pour la détection d'intrusions. Nous présentons aussi notre outil de formatage, la construction des connexions, les deux types de formatage (formatage hors ligne et formatage en

temps réel). Ce chapitre présente également des résultats expérimentaux concernant le détection d'intrusions. Nous décrivons, dans un premier temps, les données que nous avons formatées avec notre outil de formatage. Par la suite, nous présentons les résultats expérimentaux concernant la détection avec des connexions finies et la détection avec des connexions non finies.

Le chapitre 8 décrit notre approche de corrélation d'alertes. Nous présentons d'abord l'extension du langage de la logique $\mathcal{MQCL}$ en un fragment de la logique du premier ordre. Par la suite, nous présentons en détail les éléments qui composent notre approche. Nous donnerons également un exemple détaillé montrant le fonctionnement de cette approche. Nous terminons ce chapitre par des résultats d'application sur les données DADDi et DARPA'99.

Nous terminons cette thèse en présentant les conclusions de nos travaux et nos perspectives.

Première partie

Modélisation des préférences

Introduction à la partie I

La notion de préférence est importante du fait que l'on se retrouve souvent dans des situations de choix parmi un ensemble d'objets à comparer ou à évaluer. C'est pourquoi, la représentation et la modélisation des préférences sont considérées comme un problème important sur lesquels de nombreux chercheurs de différents domaines se sont intéressés.

Dans cette partie, nous abordons dans quatre chapitres différents, le problème de la représentation des préférences. Les deux premiers chapitres proposent un état de l'art et les chapitres 3 et 4 présentent nos principales contributions. Nous présentons dans le premier chapitre les principales méthodes de représentation et modélisation des préférences telles que les méthodes quantitatives (numériques) et qualitatives (logiques). Dans le deuxième chapitre, nous nous intéressons à une approche récente, permettant de représenter d'une manière très simple les préférences, cette logique est appelée *logique du choix qualitatif ($\mathcal{QCL}$)*. Nous présentons les concepts essentiels de cette logique tels que le langage utilisé, la relation d'inférence, la notion d'optionalité, etc. Le troisième chapitre présente nos principales contributions qui se focalisent sur la logique du choix qualitatif. Cela est dû au fait que, d'une part, cette logique offre plusieurs avantages pour sa simplicité d'utilisation et, d'autre part, présente plusieurs limitations à cause de sa relation d'inférence qui n'est pas totalement satisfaisante, notamment en présence de la négation dans les préférences exprimées. Notre travail dans les chapitres 3 et 4 consiste donc à traiter particulièrement les limites de cette logique et à proposer de nouvelles logiques basées sur de nouvelles définitions de la négation, conjonction et disjonction des préférences.

Chapitre 1

Modélisation des préférences : État de l'art

Sommaire

1.1 Introduction

Face aux problèmes quotidiens, on se retrouve souvent confronté à des situations où la prise de décisions est nécessaire. Prendre une décision consiste à agir en fonction de plusieurs éléments à savoir nos objectifs, nos choix, nos préférences, ainsi que nos croyances et connaissances sur le monde. Dans différentes disciplines, dès lors qu'on se préoccupe de décision, il est naturel de chercher à modéliser comment comparer en terme de préférences les objets concernant la décision à prendre, c'est la raison, pour laquelle, différentes disciplines telles que l'économie (par exemple [Fishburn 1970, Armstrong 1939, Armstrong 1948, Debreu 1954]), la psychologie (par exemple [Coombs & Smith 1973, Chisholm & Sosa 1966, Kahneman & Tversky 1979]), la philosophie (par exemple [von Wright 1963]), les sciences politiques (par exemple [Sen 1986]), l'intelligence artificielle (par exemple [Wellman & Doyle 1992]) ont donné naissance à de nombreux formalismes (logiques, graphiques ou autres modes de représentation) permettant d'exprimer les préférences.

Ainsi, la notion de préférence est introduite lorsqu'un agent se retrouve face à un choix parmi un ensemble d'alternatives. Il est donc nécessaire de pouvoir évaluer chacune des alter-

natives concernées tout en considérant des informations dont un agent dispose (comme par exemple le type de résultat attendu). A titre d'exemple, supposons qu'un agent doit acheter un billet d'avion le plus rapidement possible pour une mission de travail à l'étranger. Une agence de voyage ou un site électronique peut lui offrir plusieurs possibilités (alternatives). Chaque alternative est caractérisée par différents "critères" à savoir le prix, l'horaire, le confort (première ou seconde classe), avec ou sans escale, etc. Si l'agent préfère un voyage moins cher, sans escale, plus confortable alors différentes méthodes peuvent être appliquées pour satisfaire ses préférences. Une première procédure toute simple consiste à éliminer certaines alternatives (alternatives non-buts ou non satisfaisantes) et considérer d'autres alternatives meilleures (alternatives satisfaisantes), il s'agit donc d'identifier un sous ensemble aussi restreint que possible, composé des alternatives jugées comme les meilleures. C'est ce que l'on appelle *problématique de choix* dans un problème de décision [Roy 1985]. Une deuxième possibilité concerne le cas où chaque alternative peut être considérée indépendamment des autres en la positionnant sur une échelle ou en la comparant à des normes ou des niveaux de référence (chaque alternative sera classée dans une classe prédéfinie). Il s'agit de la *problématique de tri*. Une autre possibilité concerne la situation de comparer les alternatives entre elles, afin d'établir un classement constituant un ordre partiel. Le classement de l'ensemble des alternatives de la plus mauvaise à la meilleure, représente le niveau de satisfaction de chaque alternative, il s'agit ici de la *problématique de rangement*.

C'est ainsi donc que les alternatives sont évaluées de différentes manières selon la nature des jugements apportés (absolus ou relatifs). Concernant la problématique de choix et la problématique de rangement, les alternatives sont évaluées à base des jugements comparatifs ou relatifs. Il s'agit d'une *évaluation relationnelle, logique ou qualitative*. Lorsqu'il s'agit de la problématique de tri, l'évaluation des alternatives se base sur des jugements absolus. Il s'agit donc d'une *évaluation quantitative*.
Trois types de préférences se distinguent selon la façon avec laquelle les alternatives concernées sont évaluées.

- **Préférences tout ou rien** : lorsque les mauvaises alternatives sont écartées et celles jugées meilleures sont considérées, autrement dit lorsque l'évaluation des alternatives est faite à base de jugements comparatifs ou relatifs, on parle de préférences *tout ou rien*. Ces préférences sont peu expressives puisqu'elles ne permettent pas de constituer une forme de gradualité (intensité de préférence) dans la satisfaction des objets [Lang 2003].

- **Préférences relationnelles** : lorsque les différentes alternatives constituent un préordre éventuellement partiel à base des jugements comparatifs ou relatifs, il s'agit des *préférences relationnelles*. Ce type de préférences ne met en évidence que des alternatives de niveaux de satisfaction différents, sans que l'on puisse quantifier les intensités des préférences. Les préférences constituent donc la gradualité sans l'intensité.

- **Préférences cardinales** (ou *quantitatives*) : lorsque des valeurs intrinsèques sont affectées aux alternatives (l'évaluation fait appel à des jugements absolus), il s'agit des *préférences cardinales*.

Les deux premiers types de préférences sont souvent qualifiées de représentations *qualitatives* et les préférences concernant le troisième type sont qualifiées de représentations *quantitatives*.

1.2 Modélisation des préférences

Les alternatives sur lesquelles des agents peuvent définir leurs préférences sont souvent très nombreuses et de nature hétérogènes. Cela pose un problème majeur pour la représentation des préférences, du fait que le nombre de solutions peut exploser. C'est pourquoi, des méthodes automatisées telles que des approches logiques sont nécessaires. Dans ce qui suit, nous introduisons brièvement, la représentation relationnelle (qualitative) des préférences, ainsi que la représentation numérique (quantitative).

1.2.1 Représentation qualitative des préférences

Dans un langage courant, préférer un objet à un autre du point de vue de la satisfaction qu'il procure, revient souvent à utiliser des expressions de comparaison comme "*meilleur que, pire que, aussi bon que, indifférent à*". Considérons a_1 et a_2 deux alternatives à comparer, la réponse à la question : "*l'alternative* a_1 *est-elle-au moins aussi bonne que l'alternative* a_2 *?*", peut être décrite par une relation binaire entre ces deux alternatives.

Une relation binaire $\mathcal{R}$ sur un ensemble d'alternatives $\mathcal{A}$ est un sous ensemble de produit cartésien $\mathcal{A} \times \mathcal{A}$. L'alternative a_1 est donc en relation avec l'alternative a_2 si le couple (a_1, a_2) appartient à $\mathcal{R}$. Nous noterons fréquemment $a_1\ \mathcal{R}\ a_2$ au lieu de $(a_1,\ a_2) \in \mathcal{R}$.
Nous présentons, ci-dessous, quelques propriétés fréquentes d'une relation binaire $\mathcal{R}$ ([Monjardet 1978, Bouyssou & Vincke 2003]). Notons qu'une relation binaire peut vérifier (ou ne pas vérifier) de nombreuses propriétés.

Une relation binaire $\mathcal{R}$ sur l'ensemble d'alternatives $\mathcal{A}$ est dite :
- réflexive : $\forall\ a_1 \in \mathcal{A},\ a_1\ \mathcal{R}\ a_1$,
- irréflexive : $\forall\ a_1 \in \mathcal{A}$, $(a_1\ \mathcal{R}\ a_1)$ n'est pas vrai,
- asymétrique : $\forall\ a_1,\ a_2 \in \mathcal{A}$, si $(a_1\ \mathcal{R}\ a_2)$ alors $(a_2\ \mathcal{R}\ a_1)$ n'est pas vrai,
- symétrique : $\forall\ a_1,\ a_2 \in \mathcal{A}$, si $(a_1\ \mathcal{R}\ a_2)$ alors $(a_2\ \mathcal{R}\ a_1)$,
- antisymétrique : $\forall\ a_1,\ a_2 \in \mathcal{A}$, si $((a_1\ \mathcal{R}\ a_2)$ et $(a_2\ \mathcal{R}\ a_1))$ alors $(a_1 = a_2)$,
- transitive : $\forall\ a_1,\ a_2,\ a_3 \in \mathcal{A}$, si $((a_1\ \mathcal{R}\ a_2)$ et $(a_2\ \mathcal{R}\ a_3))$ alors $(a_1\ \mathcal{R}\ a_3)$,
- complète : $\forall\ a_1,\ a_2 \in \mathcal{A}$, $(a_1\ \mathcal{R}\ a_2)$ ou $(a_2\ \mathcal{R}\ a_1)$.

Pour tout couple d'alternatives (a_1, a_2) dans $\mathcal{A}$, la relation de comparaison entre ces alternatives peut être interprétée comme suit :

- **Relation de préférence $\succeq$:** si l'alternative a_1 est au moins aussi préférée que l'alternative a_2, alors la relation binaire $\mathcal{R}$ entre ces deux alternatives peut être représentée par la relation de préférence $a_1 \succeq a_2$.
- **Relation de préférence stricte $\succ$:** si a_1 est strictement préférée à a_2, alors la relation binaire $\mathcal{R}$ entre ces deux alternatives peut être représentée par la relation de préférence stricte $a_1 \succ a_2$. Dans ce cas, nous avons : $a_1 \succeq a_2$ et $non(a_2 \succeq a_1)$.
- **Relation d'indifférence $\sim$:** si a_1 est indifférente à a_2, alors la relation binaire $\mathcal{R}$ entre ces deux alternatives peut être représentée par la relation d'indifférence $a_1 \sim a_2$. Dans ce cas, nous avons : $(a_1 \succeq a_2)$ et $(a_2 \succeq a_1)$.
- **Relation d'incomparabilité :** pour ce type de relation, il peut arriver que l'on ne soit pas capable de préférer une alternative à une autre. Nous avons $non(a_1 \mathcal{R} a_2)$ et $non(a_2 \mathcal{R} a_1)$.

Selon les propriétés que la relation binaire $\mathcal{R}$ vérifie (ou satisfait) sur l'ensemble des alternatives, la relation de préférence $\succeq$ peut être considérée comme un ordre total (c-à-d. relation complète, antisymétrique et transitive), un pré-ordre total (c-à-d. relation complète et transitive), un ordre partiel (c-à-d. relation réflexive, antisymétrique et transitive), pré-ordre partiel (c-à-d. relation réflexive et transitive), etc. Dans ces cas, la meilleure alternative peut toujours exister. Cependant, si aucune contrainte n'est imposée aux relations binaires, il se peut qu'il n'existe pas de meilleures alternatives, comme c'est le cas lorsque la relation est non transitive par exemple.

1.2.2 Représentation numérique des préférences

Dans ce cas, pour représenter les préférences d'un agent, une ou plusieurs valeurs numériques sont affectées à chaque alternative d'une manière à refléter l'ordre, ou le classement, que l'agent a établi entre les alternatives, ou aussi pour représenter le niveau de désirabilité des alternatives considérées. Cette représentation se base sur ce que l'on appelle *fonction d'utilité*.

On appelle *utilité*, la valeur associée à une alternative et on appelle *fonction d'utilité*, la fonction $u : \mathcal{A} \rightarrow \Re$ qui associe une utilité à chaque alternative. La fonction d'utilité se base sur un critère permettant d'attribuer une valeur plus élevée à une alternative qui est plus désirable qu'une autre (*utilité cardinale*).
Dans la théorie de l'utilité cardinale, on considère que la valeur de la fonction d'utilité d'une alternative mesure la satisfaction que l'agent tire de cette alternative. Par exemple, une alternative a ayant une valeur d'utilité deux fois plus grande qu'une autre alternative b sera deux fois plus préférée : $u(a) = 2u(b)$, avec $u(b) > 0$.

Lorsque chaque alternative est associée d'une seule valeur numérique, il s'agit d'un problème de décisions *mono-critère*. Par ailleurs, il est possible qu'une alternative sera évaluée sur plusieurs critères, c'est-à-dire que chaque alternative est associée de plusieurs valeurs numériques. Il s'agit dans ce cas d'un problème de décision *multi-critères*. Un exemple d'un problème de décision multi-critère est celui cité dans [Ehrgott 2000]. Cela concerne un choix d'une automobile, en prenant en compte les critère *"prix, consommation et puissance"* de quatre modèles d'automobiles (alternatives). Ces alternatives sont *"Citroen, Peugeot, Renault, Ford"*. Chaque alternative prend trois valeurs (une valeur pour chaque critère).

Selon [Fishburn 1999], les jugements humains (absolus ou relatifs) sont liés à la notion d'ordre formalisée par la relation de transitivité, et la décision (choix de la meilleure alternative) est associée au concept de maximisation, c'est pourquoi de nombreux chercheurs considèrent que ces deux représentations (numériques et qualitatives) sont nécessaires dans le problème de la modélisation des préférences.

Dans les sections suivantes, nous abordons avec plus de détails les deux approches de représentation des préférences. Nous commençons par présenter quelques approches quantitatives, plus particulièrement, le modèle de l'utilité espérée et quelques critères de décision dans l'incertain. Par la suite, nous passons en revue quelques approches qualitatives particulièrement connues en intelligence artificielle telles que les approches permettant de représenter les préférences conditionnelles.

1.3 Modélisation numérique des préférences : approches quantitatives

La représentation numérique des préférences est largement utilisée dans la littérature, notamment dans la théorie de la décision. Nous exposons dans cette section quelques travaux fondateurs dans le domaine. Plus précisément, nous présentons, en premier, quelques modèles habituellement utilisés en théorie de la décision, particulièrement le modèle de *l'utilité espérée* introduit par von Neumann et Mongenstern [von Neumann & Morgenstern 1947] qui, par la suite, a été modifié et généralisé par de nombreux chercheurs. Le modèle de *l'utilité espérée* traite le problème de la décision dans l'incertain (décision sous risque dans le cadre de von Neumann et Mongenstern et décision sous incertitude dans le cadre de Savage). Dans ces cas, l'incertitude sur les connaissances dont l'agent dispose sur l'état du monde pour définir ses préférences est modélisée dans un cadre probabiliste. Par la suite, nous nous intéressons au même problème, mais l'incertitude est modélisée dans un cadre non probabiliste. Dans ce contexte, nous présentons les critères de décision recensés dans [Luce & Raiffa 1957].

1.3.1 Le modèle de l'utilité espérée : modèle probabiliste

Une situation de décision correspond à un choix effectué par un agent parmi une liste d'alternatives possibles, où le résultat peut être, par exemple, une alternative qui a la plus grande mesure de "désirabilité" (utilité), ou celle qui est préférée à toutes les autres. Cependant, lorsqu'un agent effectue son choix, plusieurs éléments peuvent intervenir comme par exemple : ses connaissances sur l'état du monde, ses préférences sur les résultats possibles de sa décision, ses désirs ou éventuellement certaines émotions comme la tentation ou le regret et bien évidemment d'autres facteurs. Dans des situations, où l'agent a la certitude d'obtenir l'alternative qu'il a choisie, on parle de la *décision avec certitude*, qui concerne un problème de choix déterministe, c'est-à-dire qu'il n'y a qu'une seule conséquence ou état résultat pour chaque alternative. Dans le cas où les différents facteurs ne sont pas tous connus (l'agent ne connaît pas l'état exact de l'environnement), alors il s'agit d'un problème de choix sous incertitude *(décision dans l'incertain)*. Dans ces situations, les différentes alternatives parmi lesquelles l'agent va effectuer son choix peuvent conduire à différentes conséquences possibles. L'incertitude peut être due à plusieurs éléments :

- l'agent peut être en situation d'ignorance totale s'il ne dispose d'aucune information sur son environnement.
- lorsque l'agent a des informations manquantes ou qui ne sont que partiellement observées, dans ce cas ses connaissances sur l'environnement sont incomplètes.
- les connaissances de l'agent sur son environnement sont considérées imprécises dans le cas où il dispose d'informations vagues. L'agent pourra, par exemple, connaître avec précision l'intervalle auquel appartient la valeur (numérique) d'une information au lieu d'avoir une valeur unique.

Pour analyser et modéliser ce type de situations, nous considérons que les préférences d'un agent sur les différentes alternatives sont fonction des conséquences (ou effets résultants de la mise en œuvre de chaque alternative). L'existence de l'incertitude peut être modélisée dans un cadre probabiliste [Luce & Raiffa 1957] et les préférences par une fonction d'utilité (u).

Exemple 1.1
Nous illustrons ce problème de décision sous incertitude par l'exemple de Savage [Savage 1954],

l'agent a, à sa disposition, un bol avec une omelette faite avec 5 œufs, un sixième œuf dont il ignore l'état (bon ou mauvais), et une tasse vide. Supposons que l'agent doit faire un choix entre les trois alternatives suivantes :

- casser le sixième œuf directement dans le bol,
- le casser au préalable dans la tasse vide pour savoir son état (bon ou mauvais),
- jeter l'œuf sans même le casser.

Le problème de choisir entre les trois alternatives est un problème de décision sous incertitude (on ne connaît pas a priori l'état de l'œuf). Les préférences de l'agent correspondent aux conséquences possibles de la décision selon l'état de l'œuf (résultats de la mise en œuvre de chaque alternative suivant l'état de l'œuf). Les croyances, a priori, de l'agent sur l'état de l'œuf et de l'effet procuré (plaisir ou désagrément qui lui est procuré par chacune des conséquences possibles) permettent de déterminer la meilleure alternative qu'il préfère. Ce problème de la décision avec incertitude, peut être formalisé par la *matrice de décision* qui est une représentation fréquemment utilisée dans les problèmes de décision.

	États de l'œuf	
Alternatives	œuf bon	œuf mauvais
casser l'œuf dans le bol	omelette à six œufs	omelette gâchée
casser l'œuf à part	omelette à six œufs et une tasse à laver	omelette à cinq œufs et une tasse à laver
jeter l'œuf	omelette à cinq œufs (1 œuf gâché)	omelette à cinq œufs

TAB. 1.1: Exemple de représentation d'un problème de décision avec une matrice de décision

Dans tout ce qui suit, nous considérons les notations suivantes :

- $\mathcal{S}$ représente l'ensemble des états du monde possibles. Un état du monde peut être vu comme une description complète de l'environnement (toutes les représentations possibles de l'incertitude).
- $\mathcal{A}$ correspond à l'ensemble des alternatives sur lesquelles un agent définit ses préférences.
- $\mathcal{X}$ un ensemble contenant toutes les conséquences possibles générées par la mise en œuvre de chaque alternative, étant donné un état du monde.
- Étant donné $s \in \mathcal{S}$, et l'alternative $a \in \mathcal{A}$, $a(\mathcal{S})$ est un vecteur qui précise ce que l'agent obtient dans chaque état s'il choisit cette alternative. $a(s)$ correspond à la conséquence de l'alternative a dans l'état s.
- $\{p_1, p_2, ..., p_n\}$ indiquent les probabilités d'obtenir les conséquences x_i dans $\mathcal{X} = \{x_1, ..., x_n\}$ à l'exécution des alternatives a_i.

Dans l'exemple 1.1, l'ensemble des états du monde $\mathcal{S} = \{s_1, s_2\}$ représente l'état de l'œuf : s_1 = œuf bon, s_2 = œuf mauvais.

L'ensemble d'alternatives est $\mathcal{A} = \{a_1, a_2, a_3\}$: a_1 = casser l'œuf dans le bol, a_2 = casser l'œuf à part, a_3 = jeter l'œuf.
L'ensemble de conséquences est $\mathcal{X} = \{x_1, x_2, x_3, x_4, x_5, x_6\}$:

- x_1 = omelette à six œufs, correspond à la conséquence obtenue lorsque l'alternative a_1 est choisie et l'état de l'œuf est s_1 (bon). Nous pouvons donc la représenter par $a_1(s_1)$,
- x_2 = omelette gâchée. Représentée aussi par $a_1(s_2)$, correspond à la conséquence obtenue lorsque l'alternative a_1 est choisie et l'état de l'œuf est s_2 (mauvais),
- x_3 = omelette à six œufs et une tasse à laver $(a_2(s_1))$,
- x_4 = omelette à cinq œufs et une tasse à laver $(a_2(s_2))$,
- x_5 = omelette à cinq œufs + 1 œuf gâché $(a_3(s_1))$,
- x_6 = omelette à cinq œufs $(a_3(s_2))$.

Utilité espérée (modèle probabiliste de base)

Le modèle d'utilité espérée est introduit par von Neumann et Mongenstern [von Neumann & Morgenstern 1947] comme un élément fondamental dans les théories de décision et de choix social, dans la mesure où il fournit une représentation très simple des préférences. Il est considéré comme un critère d'évaluation et de comparaison des différentes alternatives. En effet, le choix entre deux alternatives, a priori différentes, se ramène à la comparaison de deux nombres, représentant l'utilité moyenne ou l'utilité espérée, de celles-ci. Le choix de la meilleure alternative dans ce cas (pour l'agent) peut être, par exemple, une alternative ayant la meilleure conséquence possible, la moins mauvaise ou encore celle qui a, en moyenne la meilleure conséquence possible.

Du point de vue de von Newmann et Mongenstern, le modèle de l'utilité espérée est basé sur des axiomes qui ont été, à l'origine, formulés dans le cas où la distribution de probabilité des états du monde possibles est connue (probabilités objectives, indépendantes de l'agent). Dans ce cas, le choix d'un agent est, généralement, considéré sous *risque*.

A chaque alternative a dans $\mathcal{A}$ de conséquences x_1 avec probabilité p_1, x_2 avec probabilité p_2,..., x_n avec probabilité p_n, est associée une mesure appelée *utilité espérée, notée* $UE(a)$. $UE(a)$ est calculée par :

$$UE(a) = \sum_{i=1}^{n} p_i * u(x_i)^1 \tag{1.1}$$

Ainsi, si un agent doit choisir entre l'alternative a_1 de conséquences x_1 avec probabilité p_1, x_2 avec probabilité p_2,..., x_n avec probabilité p_n et l'alternative a_2 de conséquences y_1 avec probabilité p'_1, y_2 avec probabilité p'_2,..., y_m avec probabilité p'_m, alors l'alternative a_1 est préférée à l'alternative a_2 si et seulement si l'utilité espérée de a_1 est supérieure ou égale à celle de l'alternative a_2. Formellement, on écrit :

$$\forall a, b \in \mathcal{A} \quad a \succeq b \quad si \quad et \quad seulement \quad si \quad \sum_{i=1}^{n} p_i * u(x_i) \geq \sum_{j=1}^{m} p'_j * u(y_j) \tag{1.2}$$

[1] Pour des raisons de simplicité, nous avons choisi la notation p_i au lieu de $p_i(a)$

Par la suite, Savage [Savage 1954] a étendu et a modifié le modèle de l'utilité espérée dans le sens où les agents n'ont plus à choisir entre des alternatives aléatoires comme chez von Neumann et Mongenstern, mais plutôt, entre des alternatives dont les conséquences dépendent de la réalisation de l'état du monde. Dans ce cas, les probabilités des différents états du monde ne sont pas, a priori, données (probabilités subjectives ; chaque agent peut avoir une distribution de probabilité différente pour un même problème). Si un agent veut faire un calcul d'utilité espérée, il est donc nécessaire de faire apparaître des croyances probabilistes à partir de ses préférences. Ainsi, la construction de Savage propose une axiomatique sur les préférences des agents qui va permettre d'intégrer, d'une part, une fonction d'utilité et d'autre part, des croyances subjectives. Dans le cadre de Savage, l'utilité espérée d'une alternative a, étant donnés les différents états du monde possibles $s \in \mathcal{S}$ est définie de la manière suivante :

$$UE(a) = \sum_{s \in \mathcal{S}} p(s) * u(a(s)) \tag{1.3}$$

Ainsi, si un agent effectue son choix entre deux alternatives a et b étant donnés les états du monde possibles $s \in \mathcal{S}$, alors l'alternative a est préférée à l'alternative b si et seulement si l'utilité espérée de a est supérieure ou égale à celle de l'alternative b. Formellement, on écrit :

$$\forall a, b \in \mathcal{A} \quad a \succeq b \quad si \quad et \quad seulement \quad si \quad \sum_{s \in \mathcal{S}} p(s) * u(a(s)) \geq \sum_{s \in S} p(s) * u(b(s)) \tag{1.4}$$

La construction de l'utilité espérée de von Neumann et Mongestern est différente de celle de Savage du fait que, les croyances sont subjectives pour Savage, mais objectives pour von Neumann et Mongestern. Hormis cette différence, les deux modèles sont particulièrement importants dans la mesure où ils fournissent une représentation très simple des préférences (utilité espérée calculable très simplement). En outre, les deux modèles sont complémentaires dans le sens où, les préférences d'un agent qui peuvent être représentées par une telle forme de fonction d'utilité (dans le cas des deux modèles) vérifient essentiellement les propriétés de complétude, de transitivité, de continuité faible et d'indépendance, définies comme suit :

- **Complétude** : la propriété de complétude signifie que toutes les alternatives sont comparables deux à deux. La relation de préférence $\succeq$ sur l'ensemble des alternatives est complète.

- **Transitivité** : la propriété de transitivité recouvre le fait que si l'alternative a_1 est préférée à l'alternative b_1 et si b_1 est préférée à l'alternative c_1, alors a_1 est préférée à c_1. Si les deux premières propriétés (complétude et transitivité) sont vérifiées, alors $\succeq$ est un préordre complet.

- **Continuité** : la propriété de continuité signifie que si l'alternative a_1 est préférée à l'alternative b_1 et si b_1 est préférée à c_1, alors il existe un nombre réel α appartenant à l'intervalle [0, 1] tel que l'alternative qui consiste à mettre en œuvre l'alternative a_1 avec une probabilité α ou l'alternative c_1 avec la probabilité (1- α) soit indifférente à

l'alternative b_1.

La relation d'indifférence ($\sim$) entre ces deux alternatives est aussi appelée *axiome de réduction d'alternatives composées.*

- **Indépendance (ou principe de la "chose sûre" pour Savage[2])** : la propriété d'indépendance signifie que si deux alternatives a_1 et b_1 ont une partie en commun (Sure Thing Principle), et si uniquement cette partie est changée d'une manière équivalente dans les deux alternatives, alors le choix entre ces deux alternatives ne change pas.

Signalons que beaucoup de généralisations et de modifications du modèle de l'utilité espérée ont été proposées. Nous citons par exemple, le modèle d'utilité dépendant du rang [Quiggin 1982] qui généralise le modèle de von Neumann et Morgenstern dans le sens où le principe de la chose sûre (propriété d'indépendance) n'est pas exigé. Dans [Itzhak & David 1989], Gilboa et Schmeindler ont étendu le modèle de l'utilité espérée au cas où l'ensemble de croyances ($\mathcal{C}$) formulées par les agents sont décrites, par un ensemble de probabilités subjectives et non pas par une probabilité unique. L'agent procède d'abord à calculer l'utilité espérée d'une alternative (il en existe plusieurs) par rapport à toutes les probabilités dans l'ensemble de croyances ($\mathcal{C}$). Ensuite, l'évaluation de toute alternative se base sur le choix de l'utilité moyenne la plus petite au vu de l'ensemble de croyances (l'agent retient le pire des cas). Dans le cadre de ces modèles, un agent préfère l'alternative a à l'alternative b si et seulement si : $min_{p\in\mathcal{C}}\sum_{s\in\mathcal{S}}p(s) * u(a(s))$ $\geq min_{p\in\mathcal{C}}\sum_{s\in\mathcal{S}}p(s) * u(b(s))$. Pour d'autres travaux, voir : [Allais 1953, Keeney & Raiffa 1976, Choquet 1953, Ghirardato *et al.* 2004, Klibanoff *et al.* 2005, Mousseau 2003], etc.

1.3.2 Les critères de décision dans l'incertain : modèles non probabilistes

Dans les travaux que nous venons de citer, l'incertitude sur l'ensemble des états du monde possibles est modélisée dans un cadre probabiliste. Par ailleurs, il existe plusieurs autres travaux où l'incertitude est représentée dans un cadre non probabiliste, nous citons, par exemple, les travaux présentés par Luce et Raiffa [Luce & Raiffa 1957]. Ils décrivent plusieurs critères de décision dans l'incertain : *maximin*, *minimax regret*, *Hurwicz* et *Laplace.*

Pour présenter les résultats de chacun des quatre critères, nous considérons la matrice de décision de l'exemple de (Luce et Raiffa [Luce & Raiffa 1957]) présentée dans le tableau 1.2.

	s_1	s_2	s_3
a_1	2	12	-3
a_2	5	5	-1
a_3	0	10	-2

TAB. 1.2: Matrice de décision de l'exemple de Luce et Raiffa [Luce & Raiffa 1957]

Cette matrice représente un problème de décision concernant un choix à faire entre trois alternatives a_1, a_2 et a_3. Les états du monde possibles sont s_1, s_2, s_3. La mesure 2 (resp. 12, -3, 5, etc.) représente l'utilité de la conséquence obtenue du choix de l'alternative a_1 dans l'état du

[2]Sure Thing Principle.

monde s_1 (resp. a_1 dans l'état s_2, a_1 dans l'état s_3, a_2 dans l'état s_1, etc.). Nous avons donc : $u(a_1(s_1)) = 2$, $u(a_1(s_2)) = 12$, $u(a_1(s_3)) = -3$, $u(a_2(s_1)) = 5$, etc.

Le critère maximin

Ce critère est appelé aussi critère de Wald [Wald 1950]. Les alternatives sont ordonnées selon leur conséquences ayant la plus petite utilité, c'est-à-dire que chaque alternative est évaluée d'abord par rapport à l'utilité de chacune de ses conséquences comme suit :

$$u_W(a_{a\in\mathcal{A}}) = min_{s\in\mathcal{S}} u(a(s)) \tag{1.5}$$

$u_W(a)$ indique l'utilité de l'alternative a lorsque nous appliquons le critère *maximin (Wald)*. Une fois toutes les alternatives sont évaluées, la meilleure est donc celle qui satisfait :

$$max_{a\in\mathcal{A}}(u_W(a)) \tag{1.6}$$

Dans notre exemple, l'application de l'équation 1.5, donne :

- $u_W(a_1) = \min(u(a_1(s_1)), u(a_1(s_2)), u(a_1(s_3))) = -3$,
- $u_W(a_2) = \min(u(a_2(s_1)), u(a_2(s_2)), u(a_2(s_3))) = -1$,
- $u_W(a_3) = \min(u(a_3(s_1)), u(a_3(s_2)), u(a_3(s_3))) = -2$.

L'application de l'équation 1.6 donne : $max_{a_1,a_2,a_3\in\mathcal{A}}(u_W(a_1), u_W(a_2), u_W(a_3)) = -1$. Donc, la meilleure alternative est a_2. L'ordre de préférences entre ces trois alternatives est :
$a_2 \succ a_3 \succ a_1$.

Le critère *maximin* peut conduire à une mauvaise utilisation de l'information, comme nous pouvons l'observer dans l'exemple suivant :

	s_1	s_2
a_1	1	3
a_2	150	1

Les deux alternatives a_1 et a_2 sont jugées équivalentes ($u_W(a_1) = u_W(a_2)$) alors que l'alternative a_2 semble bien meilleure car $u(a_2(s_1)) = 150$. En fait, ce critère peut s'appliquer dès lors que l'ensemble de conséquences est ordonné. C'est pourquoi, il existe plusieurs travaux qui ont proposé de raffiner ce critère. Nous citons par exemple, le travail de Fargier, Lang et Schiex dans [Fargier *et al.* 1993] où, ils ont proposé l'ordre *discrimin* qui permet de comparer les alternatives dont les conséquences les pires sont différentes.

Le critère minimax regret

Le principe de ce critère [Savage 1951] consiste à minimiser le regret le plus élevé lorsqu'un agent effectue ses choix sur un ensemble d'alternatives. Les utilités des alternatives sont transformées en regrets. Le regret associé à une alternative dans un état du monde donné, correspond à la différence entre l'utilité de la meilleure conséquence qui aurait pu être obtenue à l'exécution de l'alternative pour cet état et l'utilité de l'alternative sélectionnée. La meilleure alternative est

donc celle qui satisfait :

$$min_{a\in\mathcal{A}}(max_{s\in\mathcal{S}}R(a(s))) \tag{1.7}$$

$R(a(s))$ correspond au regret associé à la réalisation de l'alternative a dans l'état du monde s.

$R(a(s))$ est calculé comme suit :

$$R(a(s)) = (max_{b\in\mathcal{A}}u(b(s))) - u(a(s)) \tag{1.8}$$

Si nous considérons le problème représenté par la matrice de décision du tableau 1.2 alors les regrets associés aux alternatives dans les états du monde possibles sont représentés dans le tableau suivant :

	s_1	s_2	s_3
a_1	**3**	0	2
a_2	0	**7**	0
a_3	**5**	2	1

Nous expliquons comment nous avons obtenu $R(a_1(s_1))$ par exemple : l'application de l'équation 1.8 donne : $R(a_1(s_1)) = (max_{a_1,a_2,a_3\in\mathcal{A}}(u_1(s_1), u_2(s_1), u_3(s_1)))$ - $u_1(s_1)$ = max(3, 0, 5) - 2 = 3. Le regret maximal pour a_1 (resp. a_2, a_3) est égal à 3 (resp. 7, 5). La meilleure alternative est a_1 qui minimise le regret maximal. Les alternatives sont classées comme suit : $a_1 \succ a_3 \succ a_2$.

Le critère Hurwicz

Ce critère prend en compte un coefficient de pessimisme $\alpha \in [0,1]$ pour faire un compromis entre l'utilité obtenue dans le pire cas et l'utilité obtenue dans le meilleur cas. La meilleure alternative est celle qui satisfait :

$$max_{a\in\mathcal{A}}(\alpha * min_{s\in\mathcal{S}}u(a(s)) + (1-\alpha) * max_{s\in\mathcal{S}}u(a(s))) \tag{1.9}$$

Notons que lorsque $\alpha = 1$, nous retrouvons le critère maximin. Dans notre exemple, ce critère conduit à choisir l'alternative a_3 avec $\alpha = 3/4$. Les alternatives sont classées comme suit : $a_3 \succ a_1 \succ a_2$.

Le critère de Laplace

Lorsqu'un agent, qui définit ses préférences sur un ensemble d'alternatives, est dans une situation d'ignorance totale sur l'état réel du monde, alors ce critère propose d'associer à chaque alternative la moyenne des utilités de toutes ses conséquences possibles. Si $\mathcal{S} = \{s_1, ..., s_n\}$, la meilleure alternative est donc celle qui satisfait :

$$max_{a\in\mathcal{A}}(\frac{u(a(s_1)) + u(a(s_2)) + ... + u(a(s_n))}{n}) \tag{1.10}$$

Dans notre exemple, ce critère conduit à choisir l'alternative a_1. L'ordre dans lequel les alternatives sont classées est : $a_1 \succ a_2 \succ a_3$.

Malgré la diversité des critères de décisions qui peuvent être utilisés pour le choix dans l'incertitude, aucun n'est totalement satisfaisant. En fait, le choix d'un critère à utiliser, pose un grand problème du fait que pour un même problème de décision, l'utilisation de tous les critères conduisent à des solutions contradictoires. Dans notre exemple, les alternatives a_1, a_2 et a_3 sont classées dans chaque critère comme suit :

- *maximin :* $a_2 \succ a_3 \succ a_1$,
- *minimax regret :* $a_1 \succ a_3 \succ a_2$,
- *Hurwicz* $\alpha = 3/4$ *:* $a_3 \succ a_1 \succ a_2$,
- *Laplace :* $a_1 \succ a_2 \succ a_3$.

Il est clair que l'utilisation de ces critères conduit à des situations très sensibles, ce qui nécessite une axiomatisation des différents critères. Dans [Milnor 1954], Milnor a proposé dix axiomes permettant de raffiner les différents critères.

1.4 Modélisation logique des préférences : approches qualitatives

Nous avons cité dans les paragraphes précédents que, les alternatives peuvent être aussi évaluées à base des informations relatives en utilisant des fonctions de comparaisons ou logiques. L'étude de la modélisation des préférences, du point de vue logique, est plutôt explorée en intelligence artificielle, essentiellement dans différents formalismes de raisonnement, telles que les logiques préférentielles [Halldén 1957, von Wright 1963, Shoham 1987, Doyle *et al.* 1991], la logique des défauts [Reiter 1980, Brewka 1994], la logique possibiliste à base de mesures garanties [Benferhat & Kaci 2003], la théorie de décision qualitative [Fargier & Sabbadin 2005, Boutilier 1994], les sous-théories préférées de Berewka [Brewka 1989], les approches d'argumentation [Prakken & Sartor 1996, Amgoud & Dupin de Saint Cyr 2008], etc. L'objectif de cette section est de passer en revue quelques modélisations qualitatives des préférences, développées dans la communauté de l'intelligence artificielle. Nous allons nous intéresser aux formalismes suivants :

- **Logiques conditionnelles :** ces logiques permettent de représenter des préférences conditionnelles, appelées également *préférences contextuelles* qui dépendent d'un contexte donné. Dans le cas général, les préférences sont de type "dans le contexte ϕ, l'agent préfère que ψ soit vraie" (où ϕ et ψ sont des alternatives ou des propositions). Comme exemple de ces logiques, nous présenterons la logique QDT (Qualitative Decision Theory) de Boutilier [Boutilier 1994].

- **Logiques des préférences de type *Ceteris Paribus* :** les préférences représentées dans cette famille de logiques se basent sur le principe *Ceteris Paribus* [Halldén 1957, von Wright 1963, Hansson 1996, Doyle *et al.* 1991].

Ces préférences sont également considérées comme des préférences conditionnelles, sauf que la notion *"toutes choses étant égales par ailleurs"* est imposée. Dans ce contexte, nous présenterons les *CP-nets* qui sont des modèles de représentation graphique permettant d'exploiter les relations d'indépendance entre les variables sous l'hypothèse *Ceteris Paribus*.

- **Logiques pondérées :** dans ce cadre, les préférences sont codées par des formules propositionnelles pondérées. Les logiques de pénalités [Pinkas 1991] et la logique possibiliste [Dubois *et al.* 1994, Benferhat *et al.* 2001] sont des exemples de logiques permettant de représenter les préférences par des formules pondérées. Nous présenterons dans ce contexte, deux approches qualitatives dans le cadre de la logique possibiliste. La première concerne la représentation bipolaire des préférences et la seconde concerne l'extension du modèle de l'utilité espérée dans un cadre qualitatif possibiliste.

1.4.1 Logiques de représentation des préférences conditionnelles

Comme nous l'avons mentionné précédemment, ce type de logiques permet d'exprimer les préférences conditionnelles. Ainsi, nous présentons dans cette section la logique proposée par Boutilier dans [Boutilier 1994], appelée QDT (Qualitative Decision Theory). Cette logique est basée sur la sémantique des mondes possibles permettant de raisonner sur des préférences conditionnelles exprimées qualitativement pour déterminer les buts d'un agent.

La sémantique de la logique QDT

La logique QDT (Qualitative Decision Theory) [Boutilier 1994] permet à la fois de décrire des préférences conditionnelles et des règles par défaut (connaissances incertaines). Les formules du langage QDT sont construites à partir d'un ensemble de variables propositionnelles dans $PROP_{PS}$ avec les connecteurs habituels de la logique propositionnelle ($\neg$, $\wedge$, $\vee$) ainsi que quatre opérateurs modaux $\Box_P$, $\overleftarrow{\Box}_P$, $\Box_N$, $\overleftarrow{\Box}_N$. D'autres opérateurs modaux peuvent être définis par une combinaison des différents opérateurs de base :

- L'opérateur $\overleftrightarrow{\Box}_P$ (respectivement $\overleftrightarrow{\Box}_N$) est défini à partir de $\Box_P$ et $\overleftarrow{\Box}_P$ (respectivement $\Box_N$, $\overleftarrow{\Box}_N$), tel que $\overleftrightarrow{\Box}_P\alpha = \Box_P\alpha \wedge \overleftarrow{\Box}_P\alpha$ (respectivement $\overleftrightarrow{\Box}_N\alpha = \Box_N\alpha \wedge \overleftarrow{\Box}_N\alpha$).
- L'opérateur modal $\overleftrightarrow{\Diamond}_P$ (respectivement $\overleftrightarrow{\Diamond}_N$) est le dual de $\overleftrightarrow{\Box}_P$ (respectivement $\overleftrightarrow{\Box}_N$). $\overleftrightarrow{\Diamond}_P\ \alpha = \neg\overleftrightarrow{\Box}_P\neg\alpha$ (respectivement $\neg\overleftrightarrow{\Box}_N\neg\alpha$).

La sémantique de la logique QDT est basée sur la notion des modèles de la forme $M = < \Omega, \leq_P, \leq_N, \phi >$, où Ω constitue l'ensemble de tous les mondes possibles (interprétations du langage propositionnel), ϕ est une fonction de valuation qui associe toute variable de $PROP_{PS}$ à un ensemble de mondes sur laquelle elle est vraie, et $\leq_P$ est une relation binaire transitive et complète sur Ω, utilisée pour représenter les préférences de l'agent. $\leq_N$ est également une relation binaire complète et transitive sur Ω, utilisée pour représenter la normalité (les connaissances par défaut).

Dans le cas de la logique QDT, ces opérateurs sont interprétés selon les deux relations binaires définies $\leq_P$ et $\leq_N$.

$I \leq_P I'$ indique que le monde I est au moins autant préféré que I' et $I \leq_N I'$ indique que le monde I est au moins aussi normal que I'. La satisfaction d'une formule par le modèle M est donnée par :

1. $M \models_I \Box_P \alpha$ (resp. $\Box_N \alpha$) ssi $\forall\ I' \leq I$, $M \models_{I'} \alpha$. Cela signifie que la formule $\Box_P \alpha$ (resp. $\Box_N \alpha$) est vraie dans un monde I si et seulement si α est vraie dans tous les mondes au moins autant *préférés* (resp. normales) que I.
2. $M \models_I \overleftarrow{\Box}_P \alpha$ (resp. $\Box_N \alpha$) ssi $\forall I' < I$, $M \models_{I'} \alpha$. La formule $\overleftarrow{\Box}_P \alpha$ (resp. $\Box_N \alpha$) est vraie dans le monde I si et seulement si α est vraie dans tous les mondes moins *préférés* (resp. normales) que I.
3. $M \models_I \overleftrightarrow{\Box}_P \alpha$ ssi $M \models_I \Box_P \alpha$ et $M \models_I \overleftarrow{\Box}_P \alpha$. Cela signifie que la formule $\overleftrightarrow{\Box}_P \alpha$ (c-à-d. $\Box_P \alpha \wedge \overleftarrow{\Box}_P \alpha$) est vraie dans le monde I si et seulement si les formules $\Box_P \alpha$ et $\overleftarrow{\Box}_P \alpha$ sont vraies dans ce monde (α est vraie dans tous les mondes).
4. $M \models_I \overleftrightarrow{\Diamond}_N \alpha$ ssi $\exists\ I'$ tel que $M \models_{I'} \alpha$. La formule $\overleftrightarrow{\Diamond}_N \alpha$ est le dual de $\overleftrightarrow{\Box}_P \alpha$. Elle est vraie dans le monde I si et seulement si α est vraie dans au moins un monde.

Par exemple, soit A, B deux variables propositionnelles, l'ensemble des mondes possibles sont : I_1, I_2, I_3 et I_4 (voir la figure 1.1). M est un modèle composé de ces quatre mondes ordonnés comme suit : $I_1 \leq I_2 < I_3 \leq I_4$.

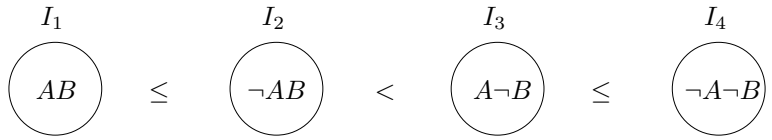

FIG. 1.1: Exemple de modèle QDT

Selon l'ordre que constituent les quatre mondes de la figure 1.1, nous avons $M \models_{I_2} \Box_P B$, car B est satisfaite dans tous les mondes autant préférés que I_2.

Pour exprimer les préférences conditionnelles (préférences relatives à un contexte), Boutilier a défini un connecteur conditionnel, qu'il a appelé *opérateur modal d'idéalité*, noté $I(-|-)$. La notation $I(A|B)$ est interprétée par "dans les mondes préférés où A est vraie, alors B est également vraie". $I(B|A)$ est définie de la façon suivante :

$$I(B|A) \equiv \overleftrightarrow{\Box}_P(\neg A) \vee \overleftrightarrow{\Diamond}_P(A \wedge \overleftrightarrow{\Box}_P(A \rightarrow B)) \tag{1.11}$$

Une *préférence relative* entre deux propositions est définie par $A \leq_P B = \overleftrightarrow{\Box}_P(B \rightarrow \Diamond_P A)$.

Cela signifie que A est, au moins, autant préférée que B si et seulement si les meilleurs A-mondes sont, au moins, aussi bons que les meilleurs B-mondes.
Une *préférence absolue* (préférence sans condition) est exprimée par la formule $I(A|T)$ ou $I(A)$ (notation abrégée). Elle s'interprète par "Idéalement A est vraie".

L'opérateur de normalité permettant d'exprimer les connaissances incertaines ou défauts est défini comme suit :

$$A \Rightarrow B \equiv \overleftrightarrow{\Box}_N \neg A \vee \overleftrightarrow{\Diamond}_N (A \wedge \overleftrightarrow{\Box}_N (A \rightarrow B)) \tag{1.12}$$

La logique QDT de Boutilier [Boutilier 1994] est intéressante du fait qu'elle donne la possibilité d'exprimer des connaissances par défaut et de définir les buts d'un agent en utilisant ses préférences étant donné un contexte. Cependant, cette approche présente une limite dans le sens où il n'est pas possible de trouver un compromis entre les préférences et les défauts car pour Boutilier un agent ne doit pas seulement agir comme si ses connaissances sont certaines, mais il doit également agir comme si, tout ce qu'il pense le plus probable (défauts) est vrai.

1.4.2 Graphe de préférences de type *Ceteris Paribus*

Nous allons présenter dans cette section, l'un des premiers travaux ayant exploité l'hypothèse *Ceteris Paribus* dans un contexte graphique. Ce modèle est appelé *CP-net*. Nous présentons en premier le principe ***Ceteris Paribus***.

Ceteris Paribus

Le principe *Ceteris Paribus* est considéré comme une hypothèse importante dans la mesure où elle permet de spécifier et représenter les préférences d'une manière intuitive. L'étude des préférences basée sur le principe *Ceteris Paribus* a été initiée par plusieurs chercheurs, en particulier, von Wright dans [von Wright 1963], Hansson dans [Hansson 1996] et Doyle, Shoham et Wellman dans [Doyle *et al.* 1991], etc.

Ceteris Paribus s'interprète par *"toutes choses étant égales par ailleurs"*. Cela signifie, pour que les alternatives considérées soient comparables entre elles, elles doivent vérifier les mêmes autres propriétés. L'exemple typique, utilisé pour illustrer le principe *Ceteris Paribus* est celui donné dans [Hansson 1996], où il exprime ses préférences sur le choix à faire entre une table ronde et une table carrée qu'il doit acheter pour son salon : "*une table ronde est préférée qu'une table carrée*". Par cette préférence, il ne veut certainement pas dire que n'importe quelle table ronde sera préférée à une table carrée, mais il veut exprimer le fait qu'il préférera toujours une table ronde à une table carrée si ces deux tables ne différent pas significativement (égales par ailleurs) dans leurs autres caractéristiques telles que la taille, le type du bois utilisé, le prix, les finitions, etc. C'est de là que vient donc l'interprétation *"toutes choses étant égales par ailleurs"*. Cette interprétation est exprimée d'une manière générale. Plusieurs autres interprétations existent désormais dans différents travaux [Lang 2003], [Tan & Pearl 1994].

Les CP-nets

Vu la taille des alternatives qui peut croître exponentiellement avec la taille des problèmes étudiés, les préférences ne peuvent pas toujours être spécifiées par un préordre sur cet ensemble d'alternatives. C'est pourquoi, exprimer par exemple les préférences via l'hypothèse *Ceteris Paribus* comme dans le cas d'un CP-net apporte certainement une grande simplicité d'utilisation en raison de la structure particulière de ces préférences. Les CP-nets (en abrégé pour Conditional

Preferences Networks ou Ceteris Paribus Networks) ont été introduits dans [Boutilier *et al.* 1999] ; ce sont des modèles graphiques permettant de représenter d'une manière simple, compacte et intuitive des préférences conditionnelles. Les préférences sont exprimées qualitativement, et sont principalement représentées par un graphe qui exploite les relations d'indépendance entre les variables du problème étudié sous l'hypothèse *Ceteris Paribus*.
Le contexte de la représentation graphique n'a jamais été fait dans le raisonnement *Ceteris Paribus*, malgré que cette hypothèse a été explorée dans de nombreux travaux en intelligence artificielle bien avant les CP-nets particulièrement dans l'un des articles fondateurs [Doyle *et al.* 1991].

Structure d'un CP-net

Un CP-net, défini sur l'ensemble des variables $V = \{X_1, X_2, ..., X_n\}$, est un *graphe orienté*, où les nœuds représentent les variables de V et les arcs représentent les relations de dépendances préférentielles entre les variables. Lorsqu'il existe un lien de la variable X à la variable Y alors X est appelé parent de Y. L'ensemble des parents d'une variable détermine les préférences sur les différentes valeurs de cette variable. Les préférences que représente un CP-net sont construites sur l'ensemble de ses alternatives. Une alternative (outcome dans [Boutilier *et al.* 1999]) est un élément du produit cartésien des domaines des valeurs des différentes variables. Au niveau de chaque nœud X_i du CP-net, les préférences sont décrites par une table de préférences conditionnelles, notée ($CPT(X_i)$) spécifiant un ordre de préférence total sur $D(X_i)$ étant donnée chaque instanciation de ses parents $Pa(X_i)$.

Pour la construction d'un CP-net, il est nécessaire d'indiquer les relations d'influence entre les préférences d'un agent, plus précisément pour chaque variable X, l'agent doit spécifier les variables qui sont susceptibles d'affecter ses préférences sur les valeurs de la variable X. Cette relation d'influence représente *l'indépendance préférentielle* entre variables qui est une notion fondamentale dans le formalisme CP-net.

Soit $V = \{X_1, X_2, ..., X_n\}$ un ensemble fini de variables. Chaque variable X_i étant associée à un domaine $D(X_i)$. Si $X = \{X_1, X_2, ..., X_m\}$ est un sous ensemble de V, alors $D(\{X_1, X_2, ..., X_m\})$ $= D(X_1) \times D(X_2) \times ... \times D(X_m)$ est le produit cartésien des domaines de chaque variable (c-à-d. valeurs possibles des variables du sous ensemble). Y est le complément de X (c-à-d. $Y = V \backslash X$). Soit $\succ$ la relation de préférence définie entre les alternatives.

Notion de dépendance préférentielle : on dit qu'un sous ensemble X est *indépendant préférentiellement* (par rapport à la relation de préférence $\succ$) de son complément Y si et seulement si, $\forall\ x_1, x_2 \in D(X)$ et $\forall y_1, y_2 \in D(Y)$ on a :

$$x_1\ y_1 \succ x_2\ y_1 \text{ si et seulement si } x_1\ y_2 \succ x_2\ y_2$$

Si cette relation est vérifiée, on dit que x_1 est *préférée Ceteris Paribus* à x_2.

Notion de dépendance préférentielle conditionnelle : on dit que X et Y sont *conditionnellement préférentiellement indépendants* (par rapport à la relation de préférence $\succ$) étant donné Z si et seulement si $\forall\ x_1, x_2 \in D(X)$, $\forall y_1, y_2 \in D(Y)$ et $\forall z \in D(Z)$ on a :

$$x_1\ y_1\ z \succ x_2\ y_1\ z \text{ si et seulement si } x_1\ y_2\ z \succ x_2\ y_2\ z$$

Exemple 1.2
Considérons l'exemple suivant qui représente les préférences d'une personne lors d'un dîner dans un restaurant. Pour des raisons de simplicité, les variables sur lesquelles portent les préférences

de cette personne sont considérées binaires. Soit donc P, V et De trois variables indiquant respectivement les préférences par rapport au : plat principal, le vin et le dessert. La personne préfère strictement que son plat principal soit à base de poisson (P_p) plutôt qu'à base de viande (P_v), tandis que ses préférences sur le vin (rouge ou blanc) dépendent de son plat principal. En effet, elle préfère du vin blanc (V_b) si son menu principal est à base de poisson, sinon elle préfère du vin rouge (V_r). Enfin, elle préfère ne pas prendre un dessert ($\neg De$) si son plat principal est accompagné du vin blanc, sinon elle préfère prendre un dessert (De). La figure 1.2 représente toutes ces informations qui décrivent les préférences de cette personne. On a donc $\mathcal{D}(P) = \{P_p, P_v\}$, $\mathcal{D}(V) = \{V_b, V_r\}$ et $\mathcal{D}(De) = \{De, \neg De\}$.

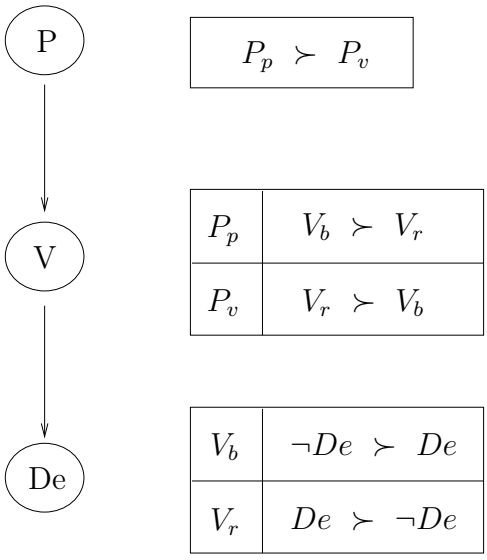

FIG. 1.2: Exemple de CP-net

La sémantique d'un CP-net

Le graphe de préférences de la figure 1.3 représente la relation de préférence induite par le CP-net de la figure 1.2. Les nœuds représentent les alternatives possibles et les arcs représentent les relations de préférences entre ces alternatives qui peuvent être déduites directement à partir des tables de préférences conditionnelles du CP-net. Par exemple, l'arc orienté du nœud qui représente l'alternative *($P_p \wedge V_b \wedge \neg De$)* vers le nœud qui représente l'alternative *($P_p \wedge V_r \wedge \neg De$)* indique que *($P_p \wedge V_b \wedge \neg De$)* est préférée à *($P_p \wedge V_r \wedge \neg De$)*. Cette préférence est déduite à partir des $CPT(V)$ et $CPT(De)$.

La sémantique d'un CP-net est définie, en terme de l'ordre, que l'ensemble des alternatives représente entre elles. Cet ordre, appelé *ordre de préférence*, correspond à l'ensemble de contraintes imposées par les tables de préférences conditionnelles du CP-net. Par exemple, l'ordre de préférence suivant (du moins préféré au plus préféré) est extrait du graphe de la figure 1.3 :

$$P_v \wedge V_b \wedge De \succ P_v \wedge V_r \wedge De \succ P_p \wedge V_r \wedge De \succ P_p \wedge V_b \wedge De \succ P_p \wedge V_b \wedge \neg De$$

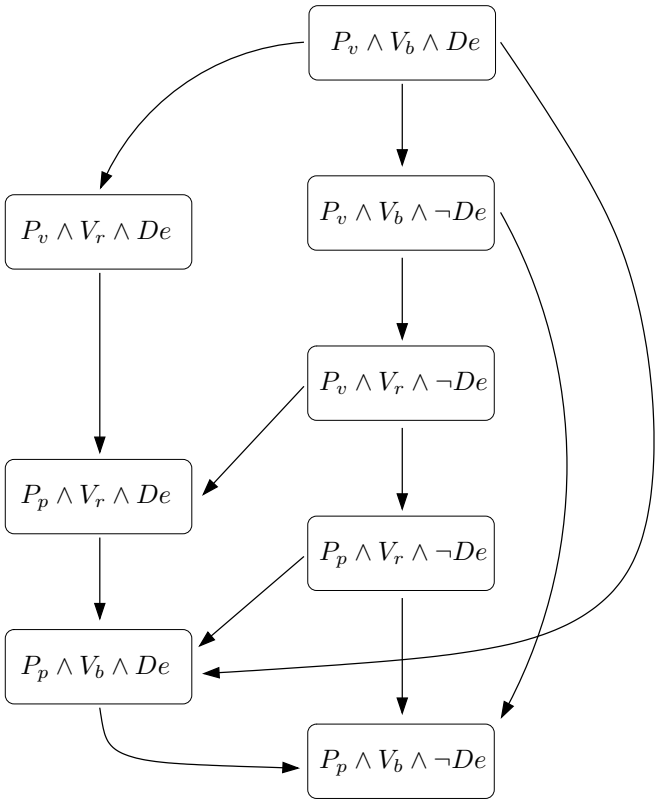

FIG. 1.3: Graphe de préférences correspondant au CP-net de la figure 1.2

Le raisonnement avec les CP-nets

La modélisation des préférences utilisant un CP-net a pour objectif de comparer les alternatives entre elles et de déterminer la meilleure (ou les meilleures s'il en existe plusieurs). Afin de comparer les alternatives entre elles, le CP-net est basé sur la notion de *changement élémentaire* (*flip*), qui consiste à changer la valeur d'une seule variable à la fois par une valeur moins préférée ou plus préférée suivant la table des préférences conditionnelles associée à cette variable. Le changement par une valeur moins préférée (resp. plus préférée) concerne un changement aggravant (resp. améliorant). Dans l'exemple 1.2, le passage de $P_p \wedge V_r \wedge \neg De$ à $P_p \wedge V_b \wedge \neg De$ est obtenu par un changement élémentaire améliorant de V_r par V_b car lorsque c'est le plat à base de poisson (P_p) qui est choisi, le vin blanc (V_b) est préféré au vin rouge V_r.
Déterminer la meilleure alternative consiste, intuitivement, à parcourir le CP-net de la racine (variable parent ou variable inconditionnelle) en feuille en positionnant la valeur des variables à la valeur préférée étant donné la valeur de ses parents. Dan notre exemple, la meilleure alternative est $P_p \wedge V_b \wedge \neg De$.

Depuis qu'il a été introduit, le formalisme CP-net est largement utilisé dans de nombreux travaux. Dans [Domshlak 2002], plusieurs analyses et extensions de ce formalisme ont été présentées. TCP-nets (tradeoffs-enhanced CP-nets) est un exemple de formalisme étendu de CP-net permettant non seulement de coder des informations sur des indépendances conditionnelles comme dans le cas d'un CP-net, mais il permet également de capturer des informations sur les conditionnels d'importance relative (conditional relative importance). Rossi dans [Rossi *et al.* 2004] a

introduit le modèle *mCP-net* qui est une extension de CP-net pour représenter les préférences conditionnelles qualitatives de plusieurs agents. Dans [Dubois *et al.* 2006], les auteurs ont proposé une approximation des CP-nets dans une logique conditionnelle des préférences, basée sur un principe d'engagement minimum dans le cadre de la logique possibiliste. Plusieurs exemples d'utilisation des CP-net ont été proposés : dans le cadre d'un problème de satisfaction de contraintes [Boutilier *et al.* 2004], dans des problèmes multi-agents [Wicker & Doyle 2007], dans le cadre des jeux booléens [Bonzon 2007], etc. Cette large utilisation s'explique d'une part, par son aspect graphique qui le rend plus simple et d'autre part du fait que les préférences sont représentées qualitativement. Cependant, ce modèle présente également quelques limites. Exemple : le langage sur lequel se base le CP-net (particulièrement lié à la notion du changement élémentaire) n'est pas totalement expressif dans la mesure où l'existence d'une suite de changements élémentaires aggravants n'implique pas forcément, l'absence d'une suite de changements améliorants. Il est, par exemple, impossible d'avoir avec un CP-net l'ordre "$P_p \wedge V_b \succ P_v \wedge V_r \succ P_p \wedge V_r \succ P_v \wedge V_b$".

1.4.3 Logiques pondérées : approches basées sur la théorie des possibilités

Dans cette section, nous nous intéressons à d'autres approches logiques de la représentation des préférences, basées sur la théorie des possibilités. Ce cadre possibiliste a connu un grand intérêt par de nombreux chercheurs dans la littérature, car il offre différents formats de représentations des préférences. Ainsi, nous pouvons citer les deux approches basées sur les utilités pessimistes et optimistes qui ont été respectivement proposées par Yager en (1979) et Whalen en (1984). En effet, ces deux utilités ont été axiomatisées dans le cadre de Von Neumann et Monrgenstern par Dubois et Prade [Dubois & Prade 1995] et par la suite dans le cadre de Savage par Dubois, Prade, Godo et Zapico [Dubois *et al.* 1998]. Dans [Benferhat *et al.* 2002], il a été proposé que les préférences d'un agent peuvent être représentées à l'aide de deux bases possibilistes différentes. L'une représente ce qui est plus au moins acceptable pour l'agent (ses rejets) et l'autre base exprime ses préférences positives.

Brefs rappels sur la théorie de possibilités

La théorie des possibilités a été introduite par Zadeh [Zadeh 1999] comme un modèle simple et naturel pour le raisonnement et le traitement des informations incertaines.
La notion de base est la *distribution de possibilité*, notée par π, qui est une fonction de l'ensemble des solutions ou interprétations dans Ω vers l'intervalle [0,1]. Intuitivement, une distribution de possibilités peut représenter soit, les connaissances que l'agent a sur les états du monde plus ou moins connues (croyances), soit ses préférences. La distribution de possibilités distingue les mondes les plus plausibles de ceux qui le sont moins ; si à chaque interprétation I est associée un degré de comptabilité avec les connaissances sur l'état du monde (resp. elle distingue également les conséquences les plus préférées de celles qui le sont moins lorsque la possibilité d'une conséquence peut être vue comme un degré auquel elle satisfait les préférences de l'agent).
Par convention, $\pi(I) = 0$ signifie que I est un état impossible, certainement pas le monde réel (resp. I est entièrement insatisfaisante si on code les préférences), alors que si $\pi(I) = 1$ signifie que I est entièrement possible ou plausible (resp. I est la solution la plus préférée et totalement satisfaisante si on code les préférences). Lorsqu'il existe une interprétation de degré de possibilité 1 (c-à-d. I est cohérente avec les informations disponibles), on dit que la distribution de possibilité est **normalisée**.

Définition 1.1
Une distribution de possibilités π est dite moins spécifique (resp. plus spécifique) qu'une distribution de possibilités π' si $\forall I \in \Omega$, $\pi(I) \geq \pi'(I)$ (resp. $\forall I \in \Omega$, $\pi(I) \leq \pi'(I)$).

La théorie des possibilités distingue deux distributions de possibilités particulières :

- La première correspond à la *connaissance complète* du monde réel : $\exists\ I_0$, $\pi(I_0) = 1$ et $\forall\ I \neq I_0$, $\pi(I) = 0$,
- la seconde correspond à *l'ignorance totale* : $\forall\ I \in \Omega$, $\pi(I) = 1$.

Dans le premier cas, l'état réel du monde est parfaitement connu, c'est-à-dire, un seul état du monde est possible et tous les autres sont impossibles. Concernant les préférences que la distribution de possibilité peut décrire dans ce cas, il n'y a qu'une seule conséquence ou état résultat pour chaque alternative (le choix entre alternatives est déterministe). Dans le second cas (ignorance totale), nous n'avons aucune information sur l'état du monde réel. Si on code les préférences, les différentes alternatives parmi lesquelles un agent va effectuer son choix peuvent conduire à différentes conséquences possibles (le choix entre alternatives est sous incertitude).

Étant donnée une distribution de possibilité, l'incertitude peut être décrite par deux mesures différentes :

1. **Mesure de possibilité** : la mesure de possibilité d'une formule ϕ, notée $\Pi(\phi)$ représente le niveau de cohérence de ϕ avec les informations représentées par π. Elle correspond au plus grand degré de possibilité associé à tous les modèles de ϕ. Formellement, nous avons :

$$\Pi(\phi) = max\{\pi(I) : I \in \Omega, I \models \phi\} \tag{1.13}$$

   La notation $I \models \phi$ signifie que I est un modèle de ϕ (un modèle est une interprétation qui satisfait une formule donnée, $\models$ est la relation d'inférence propositionnelle).
   Les conditions suivantes doivent être satisfaites par la mesure de possibilité Π, ($\top$ et $\bot$ représentent respectivement la tautologie et la contradiction) :
   - $\Pi(\top) = 1$,
   - $\Pi(\bot) = 0$,
   - $\Pi(\phi \vee \psi) = \max(\Pi(\phi),\Pi(\psi))$,
   - si ϕ et ψ ont les mêmes modèles alors $\Pi(\phi) = \Pi(\psi)$.

2. **La mesure de nécessité :** la mesure de nécessité d'une formule ϕ, notée $N(\phi)$ permet d'évaluer le degré d'inférence de ϕ à partir des informations disponibles. $N(\phi)$ représente une mesure duale de $\Pi(\phi)$:

$$N(\phi) = 1 - \Pi(\neg\phi) \tag{1.14}$$

   Cette mesure correspond au degré de certitude associé à ϕ en prenant en considération les informations dont nous disposons. Elle permet de compléter nos connaissances sur les états du monde possibles dans le sens où une mesure de possibilité ne permet pas de décrire toute l'incertitude sur le monde réel. Par exemple, le fait d'avoir $\Pi(\phi) = 1$ signifie que ϕ est entièrement possible, mais ne donne aucune information sur $\neg\phi$, en particulier $\neg\phi$ peut ne pas être exclue. Les conditions suivantes doivent être satisfaites par la mesure de nécessité N :
   - $N(\top) = 1$,
   - $N(\bot) = 0$,
   - $N(\phi \wedge \psi) = \min(N(\phi),N(\psi))$,

- si $\phi \equiv \psi$ alors $N(\phi) = N(\psi)$,
- $\forall\ \phi$, $N(\phi) \leq \Pi(\psi)$.

Exemple 1.3
Soient A et B deux variables binaires, le tableau suivant donne les degrés de possibilités associées aux différentes situations :

Interprétations $I_{i=0,...,3}$	$\pi(I_{i=0,...,3})$
AB	0.1
A¬B	0.3
¬AB	1
¬A¬B	0.6

Tab. 1.3: Distribution de possibilités jointe sur A et B

La distribution marginale associée à A est : $\Pi(A) = \max(\pi(AB), \pi(A\neg B) = 0.3$.
$\Pi(\neg A) = \max(\pi(\neg AB), \pi(\neg A\neg B) = 1$. Ainsi, $N(A) = 1$ - $\Pi(\neg(A)) = 0$.

Bases de connaissances possibilistes

Dans le cadre de la logique possibiliste, les informations incertaines sont représentées par des *formules pondérées*.
Une formule pondérée est une paire (ϕ, α), où ϕ est une formule propositionnelle dans $PROP_{PS}$, et α est un réel dans l'intervalle [0,1] qui représente le degré de nécessité minimal associé à ϕ (c'est-à-dire $N(\phi) \geq \alpha$).
L'ensemble de formules pondérées ϕ_i avec des degrés α_i constitue une base possibiliste, notée Δ de la forme :

$$\Delta = \{(\phi_i, \alpha_i) : i = 1, ..., n\} \tag{1.15}$$

où α_i représente le niveau de priorité de ϕ_i si l'on représente les préférences d'un agent ou bien le niveau de certitude de ϕ_i si l'on représente des croyances sur les états du monde réel.
Les formules de poids 0 ne sont pas présentées, car seulement les informations qui sont quelques peu certaines sont décrites.

Représentation des préférences bipolaires

Nous rappelons brièvement l'approche proposée dans [Benferhat *et al.* 2002], où les auteurs considèrent que lorsqu'un agent exprime ce qui est plus ou moins acceptable, cela reflète directement ses rejets plus ou moins forts et quand il exprime ce qu'il veut atteindre, il s'agit de ces buts positifs. Ainsi, les rejets et les buts positifs d'un agent peuvent être représentés dans le cadre de la logique possibiliste à l'aide de deux bases différentes.

1) Représentation logique des rejets

L'ensemble des rejets d'un agent, noté R constitue un préordre total sur l'ensemble des interprétations au travers une distribution de possibilités π_R, qui est une fonction de Ω dans

l'intervalle [0,1]. $\pi_R(I)$ représente le degré d'acceptabilité de la solution I étant données les préférences de l'agent. Si $\pi_R(I) = 1$, alors la solution I est totalement acceptable, $\pi_R(I) = 0$ signifie que la solution I est rejetée. La solution I' est considérée plus inacceptable que la solution I si $\pi_R(I') < \pi_R(I)$.
Pour calculer la distribution de possibilités de l'ensemble de rejets R, on associe à chacun un poids qui reflète le niveau de chaque rejet. Cet ensemble est noté par $R = \{\mathcal{R}(\phi_i) = a_i) : i = 1, ..., n\ \}$, $\mathcal{R}(\phi_i) = a_i$ signifie que l'agent rejette la formule ϕ avec le poids a_i. L'ensemble R ne contient que des formules dont le poids de chacune est supérieur à 0. La distribution de possibilités associée à un ensemble de rejets $R = \{\mathcal{R}(\phi_i = a_i) : \text{i} = 1,...,\text{n}\}$ est définie par :

$$\pi_R(I) = 1 - max\{a_i : I \models \phi_i, \mathcal{R}(\phi_i) = a_i \in R\} \tag{1.16}$$

Exemple 1.4
Soient $\phi_1 = sc \wedge pl \wedge vl$, $\phi_2 = sc \wedge \neg\ pl \wedge \neg\ vl$ et $\phi_3 = sc \wedge \neg\ pl \wedge vl$ trois formules qui représentent les rejets exprimés par un agent tel que $R = \{\mathcal{R}(\phi_1) = 1, \mathcal{R}(\phi_2) = \frac{1}{2}, \mathcal{R}(\phi_3) = \frac{1}{4}\ \}$. Cette base décrit, respectivement, que l'agent a un rejet total pour une sortie dans la campagne (sc) à vélo (vl) s'il pleut (pl), un rejet partiel pour une sortie dans la campagne sans vélo lorsqu'il ne pleut pas et enfin, un rejet faible pour une sortie dans la campagne à vélo quand il ne pleut pas. La distribution de possibilité associée à R est donnée dans le tableau suivant :

I	$\pi_R(I)$
$\neg\ sc \wedge pl \wedge vl$	1
$\neg sc \wedge pl \wedge \neg\ vl$	1
$\neg\ sc \wedge \neg pl \wedge vl$	1
$sc \wedge \neg pl \wedge \neg vl$	$\frac{1}{2}$
$sc \wedge \neg pl \wedge vl$	$\frac{3}{4}$
autres interprétations	0

TAB. 1.4: La distribution de possibilités associée à l'ensemble des rejets (R)

2) Représentation logique des buts positifs

Les buts positifs (B) exprimés par un agent peuvent être également représentés à l'aide d'une distribution de possibilités notée π_B. Par convention, $\pi_B(I) = 1$ signifie que l'agent est totalement satisfait, $\pi_B(I) = 0$ signifie que l'agent est indifférent par rapport à I.
La solution I est dite *plus satisfaisante* que I' si $\pi_B(I) > \pi_B(I')$. Pour fournir toute la distribution de possibilité, les préférences de l'agent sont plutôt représentées par des formules pondérées en terme d'une *mesure de possibilité garantie.*

Étant donnée une distribution de possibilités π, la mesure de possibilité garantie d'une formule ϕ estime le niveau de cohérence de tous les modèles de ϕ avec les informations représentées par la distribution de possibilités π.

Définition 1.2
La mesure de possibilités garantie de la formule ψ, notée $\Delta(\psi)$, est définie par : $\Delta(\psi) = min\ \{\pi(I) : I \models \psi,\ et\ I \in \Omega\ \}$

Cette fonction représente le degré de satisfaction minimal des préférences représentées par π. Par convention, nous avons $\Delta(\bot) = 1$ où $\bot$ est une contradiction.
Ainsi, les buts de l'agent qui sont représentés par $B = [\psi_j, b_j] : i = 1, ..., m\ \}$, ψ_j représente un but (ou une préférence) de l'agent et b_j est le *degré de possibilité garantie minimal* de ψ_j, c'est-à-dire que $\Delta(\psi_j) \geq b_j$. Intuitivement, toute formule [ψ_j,b_j] exprime une *contrainte positive* où toute solution qui satisfait ψ_j est considérée satisfaisante avec au moins au degré b_j. En effet, la distribution de possibilité associée à l'ensemble des buts positifs B est calculée en utilisant l'équation suivante :

$$\forall I \in \Omega, \pi_B(I) = \begin{cases} 0 & si \quad \forall [\psi_j, b_j] \in B, I \not\models \psi_j \\ max & \{b_j : \quad \forall [\psi_j, b_j] \in B, I \models \psi_j\} \quad sinon. \end{cases}$$

Exemple 1.5
Soit B un ensemble des buts exprimés par un agent tel que B = {[$sc \wedge \neg\ pl \wedge vl$, 1], [$sc \wedge \neg\ pl \wedge \neg\ vl$, $\frac{1}{2}$], [$sc \wedge pl \wedge \neg\ vl$, $\frac{1}{3}$]}. la première formule signifie que l'agent est totalement satisfait si sa sortie dans la campagne sera faite à vélo et il ne pleuvra pas. Il est moyennement satisfait si il ne pleut pas mais n'a pas de vélo et enfin il est faiblement satisfait s'il pleut, et il n'a pas de vélo. La distribution de possibilité associée à B est donnée dans le tableau suivant :

I	$\pi_B(I)$
$sc \wedge \neg\ pl \wedge vl$	1
$sc \wedge \neg\ pl \wedge \neg vl$	$\frac{1}{2}$
$sc \wedge pl \wedge \neg\ vl$	$\frac{1}{3}$
autres interprétations	0

TAB. 1.5: La distribution de possibilités associée à l'ensemble de buts positifs (B)

La représentation bipolaire des préférences telle que les auteurs l'ont proposée dans [Benferhat *et al.* 2002] est faite d'une manière indépendante, c'est-à-dire que les représentations des rejets et des buts positifs ne sont pas duales. La différence peut facilement paraître dans l'interprétation de $\pi_R(I)$ et $\pi_B(I)$. Ainsi, $\pi_R(I) = 1$ signifie que I falsifie toutes les formules de l'ensemble R, alors que $\pi_B(I) = 1$ signifie que I vérifie au moins un but totalement satisfaisant. Aussi, $\pi_R(I) = 0$ signifie que I satisfait un rejet total, $\pi_B(I) = 0$ signifie que I falsifie toutes les formules de B.

Cette approche est intéressante du fait qu'elle permet l'expression des préférences bipolaires. Cela veut dire que les agents peuvent exprimer leur préférences d'une manière à la fois partiellement positive et partiellement négative, sans ambiguïtés dans le sens où si une préférence n'est pas négative, elle n'est pas considérée positive et vice-versa. Cependant, le fait d'associer des degrés à chaque rejet ou but peut poser des problèmes, notamment lorsque le nombre d'alternatives est grand.

Modèle de l'utilité espérée dans un cadre qualitatif et possibiliste

Une contre-partie possibiliste du modèle de l'utilité espérée de Von Neumann a été proposée par Dubois et Prade [Dubois & Prade 1995]. Ce modèle se base sur les critères d'utilité pessimiste et optimiste qui ont été, respectivement, proposés par Yager en 1979 et Whalen en 1984. Contrairement au modèle probabiliste de base, où une distribution de probabilité est associée aux états du monde possibles. L'incertitude ou l'ignorance dans ce modèle se traduit par une distribution de possibilité (π) sur les états du monde possibles ($\mathcal{S}$), c'est-à-dire que les *connaissances incertaines* d'un agent sont représentées par une distribution de possibilités π sur l'ensemble des états du monde $\mathcal{S}$. Les préférences sont également représentées par une fonction d'utilité u sur l'ensemble des états du monde $\mathcal{S}$ vers une échelle totalement ordonnée et bornée. L'utilité d'une alternative a ayant pour conséquence $a(s)$ est représentée par $u(a(s))$.

- **Utilité pessimiste :** dans la formalisation pessimiste, une alternative a est évaluée de la manière suivante :

$$u_*(a) = min_{s \in S} max(1 - \pi(s), u(a(s))) \tag{1.17}$$

  Une alternative a sera mauvaise (faible utilité $u_*(a)$) dès lors qu'il existe une conséquence plausible de a ayant une utilité faible. Une alternative a sera bonne (grande utilité $u_*(a)$) si il n'existe aucune conséquence plausible de a ayant une utilité élevée.

- **Utilité optimiste :** dans la formalisation optimiste, une alternative a est évaluée de la manière suivante :

$$u^*(a) = max_{s \in S} min(\pi(s), u(a(s))) \tag{1.18}$$

  Dès lors qu'il existe une conséquence plausible d'une alternative a d'utilité élevée, alors l'utilité $u^*(a)$ de l'alternative a est élevée.

Une fois les alternatives sont évaluées, ces deux utilités sont ensuite utilisées pour les classer. Plusieurs choix sont possibles, le plus recommandé étant de classer en priorité, selon l'utilité pessimiste et d'utiliser l'utilité optimiste que pour raffinement du classement. Voir [Dubois & Prade 1995] pour plus d'informations.

1.5 Conclusion

Dans ce chapitre, nous avons présenté quelques travaux qui ont traité le problème de la représentation des préférences dans un cadre quantitatif (numérique) et qualitatif (logique).
Les formalismes basés sur les représentations quantitatives des préférences sont très utilisés dans de nombreux domaines, notamment pour les problèmes de décision dans l'incertain, et cela pour deux raisons principales :

- Possibilité de représenter d'une façon simple le niveau de désirabilité (utilité) des différentes alternatives. Le choix entre deux alternatives, a priori différentes, se ramène à la comparaison de deux nombres, représentant l'utilité espérée de celles-ci.
- La considération des situations de risque et de l'incertitude.

Nous avons présenté les formalismes qui servent de référence dans le domaine de représentation des préférences et de décision en général : le modèle de von Neumann et Mongenstern [von Neumann & Morgenstern 1947], le modèle de Savage [Savage 1954] ainsi que les critères de décisions de Luce et Raiffa [Luce & Raiffa 1957]. Toutefois, la mise en œuvre de tels formalismes fait généralement abstraction de l'aspect applicabilité et présente plusieurs inconvénients. Dans le cadre de von Neumann et Mongenstern [von Neumann & Morgenstern 1947] par exemple, les connaissances de l'agent sont représentées par des probabilités objectives (données du problème) et les préférences par l'utilité espérée. Cela n'est proprement valide que pour des environnements très particuliers du fait que les préférences n'évoluent pas de façon linéaire avec les probabilités.

Contrairement aux formalismes quantitatifs, les approches qualitatives, explorées plutôt en intelligence artificielle, offrent la facilité de représentation des préférences et l'exploitation des modèles. Les modèles développés dans cette direction, utilisent des informations naturelles, moins riches et très proches du raisonnement humain. De plus, ils peuvent traiter un nombre important d'alternatives. Pour une personne voulant, par exemple, préciser son choix entre un plat à base de poisson et un plat contenant de la viande, il est plus facile de spécifier ce choix par l'information *"poisson* préféré *à viande"* (préféré peut être un connecteur de préférence comme la disjonction ordonnée par exemple) que *"viande = 0.2, poisson = 0.6"*. Cependant, la plupart des approches qualitatives que nous avons présentées ne permettent de représenter que certaines formee de préférences, comme nous l'avons déjà vu dans le formalisme CP-net [Boutilier *et al.* 2004], qui permet de représenter uniquement des préférences conditionnelles simples, ou la méthode de Boutilier [Boutilier 1994], qui ne représente aucun compromis entre les préférences de l'agent et ses connaissances sur l'état du monde. De plus, les approches logiques que nous avons présentées, et d'autres ne permettent pas de raisonner avec les préférences, c'est-à-dire qu'elles ne donnent pas la possibilité d'exprimer par exemple des préférences de la forme *"si A est préféré à B alors C est préféré à D"* qui sont des préférences conditionnelles. C'est pourquoi, dans le cadre de notre travail, nous avons opté pour l'utilisation d'une nouvelle approche logique permettant de représenter des préférences simples ou complexes. De plus, comme, nous nous intéressons à aider un administrateur réseau à prendre des décisions sur la sécurité du système qu'il surveille, cette nouvelle approche permet de coder facilement les préférences et les connaissances de l'administrateur réseau qui concernent une très grande quantité de données réseaux. Il s'agit de la logique du choix qualitatif proposée dans [Brewka *et al.* 2004].

Chapitre 2

La logique du choix qualitatif ($\mathcal{QCL}$)

Sommaire

2.1 Introduction

La représentation des connaissances et préférences est un sujet qui a été largement exploré en intelligence artificielle [Dubois *et al.* 2002], [Boutilier 1994], [Boutilier *et al.* 1999], [Lang 1996], [Lang *et al.* 2002], [Brewka 2002]. Chacun des formalismes existants offre un cadre formel bien défini avec un mécanisme de déduction et une sémantique bien étudiée. Toutefois, certaines logiques se sont montrées insuffisantes. Elles ne permettent pas de prendre en compte le raisonnement au sujet de différents types de connaissances et préférences quelque soit leur forme ou structure. C'est pourquoi, nous trouvons plusieurs approches logiques où, malheureusement, chacune est uniquement adaptée pour la représentation des préférences d'une structure bien particulière (conditionnelles, prioritaires, uni-modales, etc.).

Dans ce chapitre, nous présentons les concepts essentiels de la logique du choix qualitatif ($\mathcal{QCL}$) [Brewka *et al.* 2004], qui est une nouvelle logique permettant d'exprimer simplement des préférences entre alternatives. Nous introduisons en particulier, la sémantique, la syntaxe et plusieurs définitions telles que la notion du degré de satisfaction d'une formule $\mathcal{QCL}$, l'optionalité, la notion d'équivalence, les modèles préférés.

2.2 Notations

Nous considérons PS un ensemble de symboles propositionnels (ou atomes) et $PROP_{PS}$ un ensemble de formules propositionnelles obtenues à partir de PS et des connecteurs habituels de la logique propositionnelle ($\wedge$, $\vee$, $\neg$). Un littéral est un atome ou la négation d'un atome.

L'équivalence logique est notée par $\equiv$. $\top$ et $\bot$ représentent respectivement la tautologie et la contradiction. Ω est l'ensemble de toutes les interprétations possibles. Une interprétation I est une affectation des valeurs de vérité classiques V, F pour les atomes propositionnels dans PS. I est représentée par l'ensemble des variables affectées à vrai. Les formules sont notées par des lettres grecques ϕ, ψ,...
Un modèle est une interprétation qui satisfait une formule donnée. La notation $I \models \phi$ signifie que I est un modèle de ϕ ($\models$ est la relation d'inférence propositionnelle).
$\vec{\times}$ est un nouveau connecteur, nommé *disjonction ordonnée*. Les formules ayant la forme $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ tel que $\forall a_i, a_i \in PROP_{PS}$ sont nommées *formules de choix de base (BCF)*. L'ensemble de formules de choix de base sont représentées par le langage BCF_{PS}. GCF_{PS} représente l'ensemble de formules qui peuvent être composées des connecteurs de la logique propositionnelle sur PS ainsi que la disjonction ordonnée. L'ensemble de formules que représente GCF_{PS} sont nommées *formules de choix générales*, notées par (GCF).

2.3 Présentation de la logique $\mathcal{QCL}$

La *logique du choix qualitatif* ($\mathcal{QCL}$), est une extension de la logique propositionnelle. La partie non-standard de QCL est un nouveau connecteur $\vec{\times}$, nommé *disjonction ordonnée*. Intuitivement, si a et b sont deux alternatives représentées comme deux formules propositionnelles, alors nos préférences sur ces deux alternatives sont codées par "$a \vec{\times} b$" qui signifie : *"a si c'est possible, mais si a est impossible alors au moins b"*. L'idée de base est :

- De $\{a \vec{\times} b\}$, on infère a (premier choix).
- De $\{a \vec{\times} b, \neg a\}$, on infère b (second choix).
- Par contre, $\{a \vec{\times} b, \neg a, \neg b\}$ est inconsistant (aucune de nos préférences n'est vérifiée).

La logique $\mathcal{QCL}$ de représentation des préférences semble être utile dans de nombreuses applications théoriques et pratiques, du fait qu'elle offre un moyen de prendre en considération les connaissances, et les préférences des agents qu'elles soient d'une forme très simple et basique ou plus complexe.

2.4 Le langage $\mathcal{QCL}$

Nous présentons dans cette section, le langage $\mathcal{QCL}$, qui est en effet composé de trois sous-langages : le langage de la logique propositionnelle, l'ensemble de formules de choix de base (BCF), et l'ensemble de formules de choix générales (GCF).

2.4.1 Les formules de la logique $\mathcal{QCL}$

Le langage $\mathcal{QCL}$ contient principalement trois types de formules : des formules propositionnelles permettant de représenter les connaissances d'un agent, des formules de choix de base pour représenter les préférences de structures simples et finalement, des formules de choix générales qui représentent les préférences de structures plus générales et plus complexes.

Les formules de choix de base (BCF)

Une formule de *choix de base* correspond à la disjonction ordonnée de deux ou plusieurs formules propositionnelles. Soit $a_1, a_2, ..., a_n$, n formules propositionnelles de $PROP_{PS}$, la formule $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ est utilisée pour exprimer et ordonner une liste d'alternatives : certaines a_i sont vraies, a_1 est préférée en premier, mais si ce n'est pas possible alors a_2, si cela n'est pas possible alors a_3, etc. L'opérateur $\vec{\times}$ est associatif, $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ peut être écrite comme $(((a_1 \vec{\times} a_2) \vec{\times} a_3) \ldots \vec{\times} a_n \ldots)))$.

Définition 2.1
L'ensemble BCF_{PS} de formules de choix de base *est défini comme suit :*

1. *Si $\phi \in PROP_{PS}$, alors $\phi \in BCF_{PS}$*
2. *Si $\phi, \psi \in BCF_{PS}$, alors $(\phi \vec{\times} \psi) \in BCF_{PS}$*
3. *Toute* formule de choix de base *est obtenue uniquement par l'application des deux règles (1) et (2).*

Les formules de choix générales (GCF)

Une formule de choix générale représente toute formule qui peut être obtenue de PS en utilisant les connecteurs $\vec{\times}$, $\wedge$, $\vee$, $\neg$. Le langage composé de *formules de choix générales* est noté par GCF_{PS}.

Définition 2.2
Le langage GCF_{PS} est défini inductivement comme suit :

1. *Si $\phi \in BCF_{PS}$ alors $\phi \in GCF_{PS}$*
2. *Si $\phi, \psi \in GCF_{PS}$ alors $(\phi \wedge \psi), \neg(\psi), (\phi \vee \psi), (\phi \vec{\times} \psi) \in GCF_{PS}$.*
3. *Toute* formule de choix générale *est obtenue uniquement par l'application des deux règles (1) et (2).*

Exemple 2.1
Supposons que deux agents expriment leurs préférences au sujet d'un voyage par *avion*. Le premier agent préfère fortement voyager avec *Air-France* et moyennement avec *KLM*. Concernant le second agent, en plus qu'il préfère *Air-France* à *KLM* comme pour le premier agent, il préfère aussi prendre la première classe (*Classe1*) à la seconde (*Classe2*).
Les préférences du premier agent sont représentées par la formule "*Air-France* $\vec{\times}$ *KLM*" qui est une formule de choix de base, et celles du deuxième agent sont représentées par la formule "(*Air-France* $\vec{\times}$ *KLM*) $\wedge$ (*Classe1* $\vec{\times}$ *Classe2*)" qui est une formule de choix générale.

Le langage BCF_{PS} permet de représenter des préférences simples. Le langage GCF_{PS} permet de représenter des préférences d'une structure simple, mais également des préférences complexes. En particulier, il permet de représenter des négations, disjonctions et conjonctions des préférences. Dans ce qui suit, nous présentons la relation d'inférence des formules de GCF_{PS} et BCF_{PS}.

2.4.2 La relation d'inférence des formules $\mathcal{QCL}$

La sémantique des formules $\mathcal{QCL}$ (BCF ou GCF) est basée sur le degré de satisfaction de chaque formule dans un modèle particulier. Comme en logique propositionnelle, une interprétation I est une affectation des valeurs de vérité classiques V, F pour les atomes propositionnels

dans PS. I est représentée par l'ensemble des variables affectées à vrai. Chaque interprétation peut satisfaire une formule avec un certain degré qui est un entier positif supérieur à 0. Dans ce qui suit, nous présentons séparément la relation d'inférence des formules de choix de base et de choix générales.

La relation d'inférence des formules de choix de base

La relation d'inférence (ou le degré de satisfaction) d'une formule de choix de base étant donnée une interprétation est définie par :

Définition 2.3 ([Brewka *et al.* 2004])
- *Soit* $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n \in BCF_{PS}$.
  $I \models_k a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ *ssi* $I \models a_1 \vee a_2 \vee ... \vee a_n$, *et* $k = min(j \mid I \models a_j)$.
- *Soit* $\phi \in PROP_{PS}$. $I \models_1 \phi$ *ssi* $I \models \phi$.

Une formule de choix de base $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ est satisfaite avec un degré k par une interprétation I si I satisfait a_k (c'est-à-dire que la k^{me} option de la formule est satisfaite) mais ne satisfait aucune des a_i pour $1 \leq i < k$ (c-à-d. les $(k-1)$ options de la formule sont falsifiées). Plus précisément, le degré de satisfaction d'une formule de choix de base $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ par l'interprétation I correspond simplement à l'entier positif k le plus petit, tel que I satisfait a_k.
Si $I \models_k a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ n'est pas satisfaite, on écrit $I \not\models_k a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$.
En particulier, s'il n'existe pas un entier k tel que $I \models_k a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ soit satisfaite, on écrira : $I \not\models a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$.

Exemple 2.2
Supposons que j'ai à choisir entre deux alternatives différentes (c'est-à-dire entre deux compagnies aériennes : *Air-France* ou *EasyJet*) où je préfère fortement voyager avec *Air-France* que *EasyJet*. La logique $\mathcal{QCL}$ permet de représenter de telles préférences par la formule de choix de base suivante :
Air-France $\vec{\times}$ *EasyJet*. *Air-France* et *EasyJet* modélisent respectivement voyager avec *Air-France*, *EasyJet*.
Soit I_1, I_2, I_3 et I_4 quatre interprétations telles que $I_1 = \{$*Air-France*$\}$, $I_2 = \{$*EasyJet*$\}$, $I_3 = \{$*Air-France*, *EasyJet* $\}$ et $I_4 = \{\emptyset\}$.

Par l'application de la définition 2.3, nous avons :
- $I_1 \models$ *Air-France*, donc $I_1 \models_1$ *Air-France* $\vec{\times}$ *EasyJet*. On dit que la formule *Air-France* $\vec{\times}$ *EasyJet* est satisfaite avec le degré 1 par l'interprétation I_1.

- $I_2 \not\models$ *Air-France*, $I_2 \models$ *EasyJet*, donc $I_2 \models_2$ *Air-France* $\vec{\times}$ *EasyJet*. La formule *Air-France* $\vec{\times}$ *EasyJet* est satisfaite avec le degré 2 par l'interprétation I_2.

- $I_3 \models$ *Air-France*, $I_3 \models$ *EasyJet*, donc $I_3 \models_1$ *Air-France* $\vec{\times}$ *EasyJet*. La formule *Air-France* $\vec{\times}$ *EasyJet* est satisfaite avec le degré 1 par l'interprétation I_3.

- $I_4 \not\models$ *Air-France*, $I_4 \not\models$ *EasyJet*, donc $I_4 \not\models Air - France \vec{\times}$ *EasyJet*. L'interprétation I_4 ne satisfait pas la formule *Air-France* $\vec{\times}$ *EasyJet*.

Pour résumer :
Air-France $\vec{\times}$ *EasyJet* est satisfaite avec le degré 1 si *Air-France* est vraie.

Air-France $\vec{\times}$ *EasyJet* est satisfaite avec le degré 2 si *Air-France* est fausse et *EasyJet* est vraie.
Air-France $\vec{\times}$ *EasyJet* est falsifiée (-) si *Air-France* et *EasyJet* sont fausses.

Air-France	*EasyJet*	*Air-France* $\vec{\times}$ *EasyJet*
F	F	-
F	V	2
V	F	1
V	V	1

TAB. 2.1: Le degré de satisfaction de la formule *Air-France* $\vec{\times}$ *EasyJet*

La relation d'inférence des formules de choix générales

Contrairement aux formules de choix de base où le degré de satisfaction est facilement calculable, la relation d'inférence des formules de choix générales nécessite d'introduire en premier la notion d'optionalité que nous définissons dans ce qui suit :

Définition 2.4
L'optionalité d'une formule $\mathcal{QCL}$ (GCF ou BCF) indique le nombre d'options qui sont possibles de satisfaire cette formule. Soit ϕ_1 et ϕ_2 deux formules dans GCF_{PS} ;
- *opt(a) = 1, a est un atome propositionnel.*
- *opt($\phi_1 \vec{\times} \phi_2$) = opt($\phi_1$) + opt($\phi_2$).*
- *opt($\phi_1 \wedge \phi_2$) = max(opt(ϕ_1), opt(ϕ_2)).*
- *opt($\phi_1 \vee \phi_2$) = max(opt(ϕ_1), opt(ϕ_2)).*
- *opt($\neg(\phi_1)$) = 1.*

L'optionalité d'une formule représente, en quelque sorte, le nombre d'alternatives concernant les préférences d'un agent. Dans l'exxemple 2.2, l'optionalité de la formule "*Air-France* $\vec{\times}$ *EasyJet*" est égale à 2 (deux alternatives sont possibles pour satisfaire les préférences exprimées : *Air-France* et *EasyJet*).

Dans ce qui suit, nous donnons la définition de la relation d'inférence des formules de choix générales, notée par $\mid\!\sim^{\mathcal{QCL}}$.

Définition 2.5 (La relation d'inférence)
Soit ϕ_1, ϕ_2 deux formules de GCF_{PS}.
1. $I \mid\!\sim_k^{\mathcal{QCL}} a$ *ssi* $k = 1$ *et* $a \in I$ *(pour un atome propositionnel a).*
2. $I \mid\!\sim_k^{\mathcal{QCL}} (\phi_1 \wedge \phi_2)$ *ssi* $I \mid\!\sim_m^{\mathcal{QCL}} \phi_1$ *et* $I \mid\!\sim_n^{\mathcal{QCL}} \phi_2$ *et* $k = max(m, n)$.
3. $I \mid\!\sim_k^{\mathcal{QCL}} (\phi_1 \vee \phi_2)$ *ssi* $I \mid\!\sim_m^{\mathcal{QCL}} \phi_1$ *ou* $I \mid\!\sim_n^{\mathcal{QCL}} \phi_2$ *et* $k = min\{r \mid I \mid\!\sim_r^{QCL} \phi_1$ *ou* $I \mid\!\sim_r^{\mathcal{QCL}} \phi_2\}$.
4. $I \mid\!\sim_k^{\mathcal{QCL}} \neg\phi_1$ *ssi* $k = 1$ *et* $\nexists\ m$ *tel que* $I \mid\!\sim_m^{\mathcal{QCL}} \phi_1$.
5. $I \mid\!\sim_k^{\mathcal{QCL}} (\phi_1 \vec{\times} \phi_2)$ *ssi* $I \mid\!\sim_k^{QCL} \phi_1$ *ou* $[I \mid\!\sim_1^{\mathcal{QCL}} \neg\phi_1$, $I \mid\!\sim_m^{\mathcal{QCL}} \phi_2$ *et* $k = m + opt(\phi_1)]$.

La notion d'équivalence entre deux formules $\mathcal{QCL}$ est définie par :

Définition 2.6
Deux formules ϕ et ψ sont dites équivalentes, noté $\phi \equiv \psi$ ssi :

1. *opt(ϕ) = opt(ψ),*
2. *et pour toute interprétation I, et entier k, nous avons $I \mid\sim_k^{\mathcal{QCL}} \phi$ ssi $I \mid\sim_k^{\mathcal{QCL}} \psi$.*

Exemple 2.3
Considérons la formule $\phi = (\textit{Air-France} \vec{\times} \textit{EasyJet}) \wedge \neg \textit{Classe1}$. ϕ est de la forme $\phi_1 \wedge \phi_2$ tel que $\phi_1 = \textit{Air-France} \vec{\times} \textit{EasyJet}$ et $\phi_2 = \neg \textit{Classe1}$.
L'optionalité de ϕ est égale à max(opt($\textit{Air-France} \vec{\times} \textit{EasyJet}$), opt($\neg \textit{Classe1}$)) = 2 (application de la définition 2.4).
Pour déterminer le degré de satisfaction de ϕ, nous appliquons l'item 2 de la définition 2.5.
Soit $I = \{\textit{EasyJet}\}$, nous avons :
$I \mid\sim_2^{\mathcal{QCL}} \phi_1$ et $I \mid\sim_1^{\mathcal{QCL}} \phi_2$, donc $I \mid\sim_k^{\mathcal{QCL}} (\textit{Air-France} \vec{\times} \textit{EasyJet}) \wedge \neg \textit{Classe1}$ et k = max(2,1) = 2.

Nous considérons dans ce qui suit, K un ensemble de formules propositionnelles qui représente les connaissances ou des contraintes d'intégrité, et T un ensemble de préférences (simples ou complexes). Pour définir la relation d'inférence entre $K \cup T$ et la formule propositionnelle ϕ, nous avons besoin de définir en premier la notion de modèles préférés.

Définition 2.7
Soit K un ensemble de formules propositionnelles et T un ensemble de préférences. L'interprétation I est un modèle de $K \cup T$ ssi

1. *I satisfait K, et*
2. *I satisfait chaque formule de T avec un certain degré, c'est-à-dire $\forall \phi \in T, \exists\ k$ tel que $I \models_k \phi$.*

Le degré de satisfaction des formules nous aide à déterminer les modèles préférés. Il existe différentes manières de le faire. Dans [Brewka *et al.* 2004], l'ordre lexicographique des modèles est utilisé. Il est basé sur le nombre de formules satisfaites avec un degré donné. L'ordre lexicographique est défini comme suit :

Définition 2.8
Soit $I^k(T)$ un sous ensemble de formules de T satisfaites par un modèle I avec un degré k. Un modèle I_1 est $K \cup T$ préféré au modèle I_2, noté $I_1 \succ_{lex} I_2$, s'il existe un entier k tel que $\mid I_1^k(T)\mid > \mid I_2^k(T)\mid$ et pour tout $j < k$: $\mid I_1^j(T)\mid = \mid I_2^j(T)\mid$.
I est un modèle préféré de $K \cup T$ ssi I est un $K \cup T$ maximum-préféré.
On dit qu'un modèle I_1 est maximum-préféré s'il n'existe pas un modèle I_2 qui est strictement $K \cup T$-préféré à I_1.

Exemple 2.4
Nous illustrons la définition des modèles préférés par cet exemple, où nous considérons que la base K contient une connaissance que nous représentons par la formule ϕ_1, tel que :

$$\phi_1 = (EasyJet \wedge Classe1) \vee (AirFrance \wedge \neg Classe1)$$

et T contient les préférences représentées respectivement par les formules ϕ_2, ϕ_3 et ϕ_4 comme suit :

$$\begin{cases} \phi_2 = AirFrance \vec{\times} EasyJet \\ \phi_3 = (\neg EasyJet \vec{\times} AirFrance) \vee Classe1 \\ \phi_4 = (AirFrance \wedge \neg Classe1) \vec{\times} (EasyJet \wedge \neg Classe1) \end{cases}$$

Pour calculer les modèles préférés de la base $K \cup T$, nous allons calculer en premier le degré de satisfaction des formules ϕ_1, ϕ_2, ϕ_3 et ϕ_4 pour toute interprétation.
Par exemple, soit $I = \{EasyJet, Classe1\}$, par l'application de la définition 2.3 pour les préférences ϕ_1, ϕ_2 et ϕ_4, nous avons :

- $I \models (EasyJet \wedge Classe1) \vee (AirFrance \wedge \neg Classe1)$, donc $I \models \phi_1$.
- $I \not\models$ AirFrance, mais $I \models EasyJet$, donc $I \models_2 \phi_2$.
- $I \not\models (AirFrance \wedge \neg Classe1)$ et $I \not\models (EasyJet \wedge \neg Classe1)$, donc $I \not\models \phi_4$.

Considérons la préférence représentée par $\phi_2 = (\neg EasyJet \vec{\times} AirFrance) \vee Classe1$. Nous avons $I \not\models (\neg EasyJet \vec{\times} AirFrance)$ et $I \models Classe1$. Nous utilisons l'item 3 de la définition 2.5, nous obtenons : $I \models_1 (\neg EasyJet \vec{\times} AirFrance) \vee Classe1$.

Donc, I n'est pas un modèle pour la base $K \cup T$ (I falsifie la formule ϕ_4), il ne peut pas donc être un modèle préféré.
Le tableau suivant donne le degré de satisfaction pour toute formule ϕ_1 de K et ϕ_2, ϕ_3 et ϕ_4 de T. Si la formule est satisfaite, le degré de satisfaction est représenté par les nombres (1, 2). Si la formule n'est pas satisfaite, cela est représenté par (-).

AirFrance	EasyJet	Classe1	ϕ_1	ϕ_2	ϕ_3	ϕ_4
F	F	F	-	-	1	-
F	F	T	-	-	1	-
F	T	F	-	2	-	2
F	T	T	1	2	1	-
T	**F**	**F**	**1**	**1**	**1**	**1**
T	F	T	-	1	1	-
T	T	F	1	1	2	1
T	T	T	1	1	1	-

TAB. 2.2: Les modèles de la base $K \cup T$

La base $\{T \cup K\}$ admet deux modèles $\{Air\text{-}France\}$, $\{Air\text{-}France, EasyJet\}$, mais $\{Air\text{-}France\}$ est le seul modèle préféré.

Nous définissons la relation d'inférence entre les formules de la base $K \cup T$ et une formule propositionnelle ϕ comme suit :

Définition 2.9
Soit K un ensemble de formules propositionnelles et T un ensemble de préférences, et ϕ soit une formule propositionnelle de $PROP_{PS}$.
$K \cup T \mid\sim \phi$ ssi ϕ est satisfaite dans tous les modèles préférés de $K \cup T$.

Exemple 2.5
Considérons la base $K \cup T$ de l'exemple 2.4 qui a $\{Air\text{-}France\}$ comme modèle préféré et soit ϕ une formule propositionnelle de $PROP_{PS}$ telle que $\phi = Air\text{-}France \wedge \neg EasyJet$. La formule ϕ est satisfaite dans le modèle préféré de la base $K \cup T$ (voir Définition 2.9), d'où $K \cup T$ satisfait ϕ. Formellement on écrit $K \cup T \mid\sim Air\text{-}France \wedge \neg EasyJet$.

2.5 Conclusion

Dans ce chapitre, nous avons présenté une logique non monotone de représentation des préférences, appelée *logique du choix qualitatif* ($\mathcal{QCL}$). Cette logique ajoute à la logique propositionnelle classique une nouvelle disjonction ordonnée symbolisée par $\vec{\times}$, qui permet d'ordonner simplement les différentes alternatives.

La logique $\mathcal{QCL}$ se montre intéressante, du fait qu'elle définit un nouveau connecteur qui permet de représenter les préférences des agents par des formules de choix de base lorsqu'elles sont d'une structure simple ou par des formules de choix générales lorsqu'elles sont plus complexes. La sémantique de $\mathcal{QCL}$ se base sur le degré de satisfaction de chaque formule étant donnée une interprétation, qui en quelque sorte représente l'ordre des alternatives parmi lesquelles l'agent effectue ses choix. Ainsi, lorsqu'il s'agit par exemple des préférences simples représentées par des formules de choix de base, alors plus les alternatives concernées sont classées ou choisies en premier, plus elles sont considérées les meilleures ou les plus préférées. En outre, un agent a la possibilité non seulement de représenter ses connaissances ou préférences d'une manière indépendante (c-à-d. chaque préférence ou connaissance est représentée par une seule formule), mais il peut également fournir un ensemble de connaissances et préférences qui seront représentées séparément dans une base de connaissances et une base de préférences. Dans ce cas, la ou les meilleure(s) alternative(s) de l'ensemble des connaissances et préférences de l'agent est (sont) trouvée (s) à l'aide des modèles préférés.

Dans le cadre de cette thèse, nous avons opté pour le choix de la logique $\mathcal{QCL}$ du fait qu'elle offre plusieurs avantages, notamment dans l'ordre de la simplicité d'utilisation et la possibilité de raisonner avec les préférences, c'est-à-dire qu'elle permet d'exprimer des préférences de la forme *"si A est préféré à B alors C est préféré à D"*. En revanche, la relation d'inférence des préférences de cette forme n'est pas satisfaisante dans le sens où la règle *"si A est préféré à B alors C est préféré à D"* est équivalente à *"si A ou B alors C est préféré à D"*. C'est pourquoi, nos principales contributions concernant la représentation des préférences se basent essentiellement sur la logique $\mathcal{QCL}$ dans le but de cerner les limites qu'elle présente et de proposer des solutions. Nous présentons les limites et les révisions de cette logique dans le chapitre suivant.

Chapitre 3

La logique du choix qualitatif : limites et révisions

Sommaire

3.1 Introduction

Les approches de modélisation des préférences existantes dans la littérature, qu'elles soient quantitatives ou qualitatives imposent souvent certaines restrictions et présentent plusieurs limites. Ces limites se manifestent souvent au niveau de la nature ou le type des préférences représentées, comme nous l'avons déjà vu par exemple dans le formalisme CP-net [Boutilier *et al.* 2004], qui permet de représenter uniquement des préférences conditionnelles simples, ou encore la méthode de Boutilier [Boutilier 1994], qui s'est avérée moins importante du fait qu'aucun compromis n'est possible entre les préférences de l'agent et ses connaissances sur l'état du monde. Ces limites et restrictions peuvent se manifester également au niveau d'autres paramètres, comme par exemple le nombre de propriétés considérées comme naturelles pour un système automatique de gestion des préférences.

La logique du choix qualitatif ($\mathcal{QCL}$), que nous avons présentée dans le chapitre précédent, présente également plusieurs limites dans le sens où sa relation d'inférence n'est pas totalement satisfaisante même, si a priori, permet de raisonner avec des préférences. En effet, supposons

que nous voulons représenter les options concernant un voyage de Paris à Toulouse. Supposons que l'agence de voyage dispose des règles suivantes *"un client qui préfère Air-France à EasyJet, achète la formule hôtel"* et *"un client qui préfère EasyJet à Air-France n'achète pas la formule hôtel"*. Quand l'agence rencontre des clients qui préfèrent *Air-France* à *EasyJet*, le comportement attendu de son système d'informations est de proposer une formule hôtel pour ces clients. La logique $\mathcal{QCL}$ ne permet pas d'inférer une telle conclusion. Nous détaillerons l'exemple plus tard. De plus, lorsque nous traitons uniquement des préférences binaires (par exemple, soit aller au cinéma ou non). Une préférence de la forme, *"je préfère plutôt aller au cinéma que de ne pas y aller"*, est codée par *"cinéma* $\vec{\times}\neg$ *cinéma"*.
Dans $\mathcal{QCL}$, la négation d'une préférence de la forme *"il n'est pas vrai que je préfère aller au cinéma à ne pas y aller"* est codée par *"*$\neg$*(cinéma* $\vec{\times}\neg$ *cinéma)"* qui est équivalente à $\perp$ (théorie contradictoire).

En fait, le traitement de la négation dans la logique $\mathcal{QCL}$ n'est pas satisfaisant [Benferhat *et al.* 2006, Benferhat & Sedki 2008b]. Dans $\mathcal{QCL}$, si nous représentons des préférences par des formules qui contiennent des négations appliquées à la disjonction ordonnée, alors ces formules sont logiquement équivalentes à des formules propositionnelles obtenues par le remplacement de la disjonction ordonnée ($\vec{\times}$) par la disjonction propositionnelle ($\vee$). Ceci représente une vraie limite de $\mathcal{QCL}$, vu que la négation a son importance pour traiter des règles de la forme "si A alors B", qui est souvent codée par $\neg A \vee b$. En effet, par exemple, $\mathcal{QCL}$ ne fait pas de distinction entre les trois règles suivantes :

- "*Air-France* $\vec{\times}$ *EasyJet* $\Rightarrow$ *Formule-hôtel*" (*une personne qui préfère Air-France à EasyJet, achète la formule hôtel*),
- "*EasyJet* $\vec{\times}$ *Air-France* $\Rightarrow$ *Formule-hôtel*" (*une personne qui préfère EasyJet à Air-France, achète la formule hôtel*), et
- "*Air-France* $\vee$ *EasyJet* $\Rightarrow$ *Formule-hôtel*" (*une personne qui voyage avec Air-France ou EasyJet, achète la formule hôtel*).

Notre objectif principal consiste d'une part à cerner les limites de la logique $\mathcal{QCL}$, et d'autre part, à proposer des solutions pour remédier aux différents problèmes posés par ces limites. Ainsi, nos propositions consistent à modifier cette logique en définissant différemment la négation dans les préférences. La modification de la négation a une importance capitale pour la représentation des préférences de la forme *"Si A est préféré à B alors C est préféré à D"*. Il existe deux façons pour résoudre ce problème. Nous proposons, en premier, une nouvelle définition de la négation des préférences, mais nous préservons les définitions de la conjonction et de la disjonction de la logique $\mathcal{QCL}$. Il s'agit d'une modification minimale de $\mathcal{QCL}$ au niveau de la négation. Nous appelons cette nouvelle logique, *Logique du Choix Qualitatif Minimale*, notée $\mathcal{MQCL}$. La nouvelle logique $\mathcal{MQCL}$ se base sur le même langage que celui de la logique $\mathcal{QCL}$, mais elle utilise des règles d'inférence différentes. En particulier, la nouvelle relation d'inférence a pour objectif de surmonter les limites de la relation d'inférence $\mathcal{QCL}$ originale. L'application de cette nouvelle logique peut être satisfaisante pour certains types de problèmes comme la corrélation d'alertes par exemple. Cependant, elle ne peut pas être appliquée à d'autres problèmes sachant que les préférences sont de plusieurs types : prioritaires, positives, etc. Ainsi, nous proposons d'apporter d'autres modifications à la logique $\mathcal{QCL}$ dans le but de pouvoir exprimer d'autres types de préférences. A ce propos, nos propositions concernent deux nouvelles logiques, $\mathcal{PQCL}$ (*Logique du Choix Qualitatif Prioritaire*) et $\mathcal{QCL}+$ (*Logique du Choix Qualitatif Positive*). Les logiques $\mathcal{PQCL}$ et $\mathcal{QCL}+$ se caractérisent par une nouvelle définition de la négation qui est, également, la même que $\mathcal{MQCL}$ et des nouvelles définitions de la conjonction et de la disjonc-

tion. Dans le cadre de la logique $\mathcal{PQCL}$, la conjonction (resp. disjonction) des préférences est prioritaire, dans le cadre de $\mathcal{QCL}+$, la conjonction (resp. disjonction) des préférences est positive.

Ainsi, la logique $\mathcal{PQCL}$ [Benferhat & Sedki 2007] permet de représenter des préférences des agents ayant différents niveaux d'importance, c'est ce que l'on appelle *préférences prioritaires*. La logique $\mathcal{QCL}+$ [Benferhat & Sedki 2008b] est adaptée particulièrement à la représentation des *préférences positives* [Dubois *et al.* 2004], plus précisément aux préférences qui permettent de représenter ce qui est souhaitable par l'agent et n'excluent jamais de solutions.

Chacune de nos logiques se caractérise par une fonction de normalisation qui permet de transformer tout ensemble de préférences, représentées par des formules de choix générales en un ensemble de préférences simplifiées, par des formules de choix de base. Représenter des préférences uniquement par un ensemble de formules de choix de base permet d'avoir des préférences en forme normale qui peuvent être utilisées dans différentes approches non monotones telles que la logique possibiliste [Dubois & Prade 2002] ou la compilation des bases de connaissances stratifiées [Darwiche & Marquis 2002, Coste-Marquis & Marquis 2004] et également pour ordonner d'une manière simple les alternatives. Dans ce chapitre, nous présentons la logique $\mathcal{MQCL}$, les logiques $\mathcal{PQCL}$ et $\mathcal{QCL}+$ seront présentées dans le chapitre suivant.

Le reste du chapitre est organisé comme suit. Nous présentons en premier, les limites de la logique $\mathcal{QCL}$ (section 3.2). Les révisions et les modifications, que nous avons apportées à $\mathcal{QCL}$ sont présentées dans la section 3.3. Par la suite, nous introduisons la première logique que nous proposons. La section 3.4 présente les principales caractéristiques de la logique $\mathcal{MQCL}$. La section 3.5 conclut le chapitre.

3.2 Les limites de la logique $\mathcal{QCL}$

Comme nous l'avons mentionné dans l'introduction, la relation d'inférence $\mathcal{QCL}$ présente quelques limites et viole quelques propriétés qui sont principalement liées à la présence de la négation dans les préférences exprimées. Dans ce qui suit, nous reprenons les trois règles suivantes, données dans l'introduction, sur lesquelles nous illustrerons les limites de la logique $\mathcal{QCL}$:

- "*Air-France* $\vec{\times}$ *EasyJet* $\Rightarrow$ *Formule-hôtel*" (*une personne qui préfère Air-France à EasyJet, achète la formule hôtel*), **(1)**
- "*EasyJet* $\vec{\times}$ *Air-France* $\Rightarrow$ *Formule-hôtel*" (*une personne qui préfère EasyJet à Air-France, achète la formule hôtel*) **(2)**
- "*Air-France* $\vee$ *EasyJet* $\Rightarrow$ *Formule-hôtel*" (*une personne qui voyage avec Air-France ou EasyJet, achète la formule hôtel*) **(3)**

3.2.1 La disjonction ordonnée et la négation

La disjonction ordonnée ($\vec{\times}$) est un connecteur de la logique $\mathcal{QCL}$ qui permet d'ordonner les alternatives. Cette disjonction est différente de la disjonction standard, dans le sens où, la disjonction standard d'un ensemble d'alternatives associe le même niveau de désirabilité à toutes les alternatives, c'est-à-dire que le choix entre une alternative et une autre est indifférent. Quant à la disjonction ordonnée, elle permet d'associer des niveaux de désirabilité différents aux alternatives concernées, le choix entre une alternative et une autre est donc censé être différent.

Or, le cadre $\mathcal{QCL}$ ne fait pas de distinction entre la disjonction standard et la disjonction ordonnée lorsque la négation est présente dans les préférences. Nous avons par exemple :

$$\neg(a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n) \equiv \neg(a_1 \vee a_2 \vee ... \vee a_n) \tag{3.1}$$

Dans notre exemple, les trois règles **(1)**, **(2)** et **(3)** données ci-dessus sont équivalentes à la formule propositionnelle : ($\neg$ *Air-France* $\wedge$ $\neg$*EasyJet*) $\vee$ *Formule-hôtel*.
Cela signifie que le choix entre *Air-France* et *EasyJet* est indifférent.
En effet, les formules $\mathcal{QCL}$ niées sont équivalentes aux formules propositionnelles, obtenues par le remplacement de la disjonction ordonnée par la disjonction standard.

3.2.2 L'incohérence dans la logique $\mathcal{QCL}$

Dans ce qui suit, nous illustrons sur un exemple d'une base de connaissances et une base de préférences que la logique $\mathcal{QCL}$ induit à une situation d'incohérence.

Exemple 3.1
Considérons l'exemple suivant où la base de connaissances K contient une connaissance que nous représentons par la formule ϕ_1, tel que :

$$\phi_1 = \neg EasyJet \vee \neg Air\text{-}France$$

et la base de préférences T contient les préférences représentées respectivement par ϕ_2, ϕ_3 et ϕ_4 comme suit :

$$\begin{cases} \phi_2 = \neg(Air\text{-}France \vec{\times} EasyJet) \vee Formule\text{-}Hotel \\ \phi_3 = \neg(EasyJet \vec{\times} Air\text{-}France) \vee \neg Formule\text{-}Hotel \\ \phi_4 = Air\text{-}France \vec{\times} EasyJet \end{cases}$$

ϕ_1 signifie que l'agence de voyage ne propose pas simultanément *EasyJet* et *Air-France* à ces clients. ϕ_2 signifie que si un client préfère *Air-France* à *EasyJet* alors l'agence lui proposera une *Formule-Hotel*, ϕ_3 signifie que si un client préfère *EasyJet* à *Air-France* alors l'agence ne lui proposera pas une *Formule-Hotel*, ϕ_4 signifie qu'un client préfère *Air-France* à *EasyJet*.

La solution attendue, est que l'agence propose la formule hôtel aux clients qui préfèrent *Air-France* à *EasyJet*. Pour cela, nous calculons les modèles préférés. Pour calculer les modèles préférés de la basse $K \cup T$ (voir les définitions des modèles préférés et la relation d'inférence $\mathcal{QCL}$ dans le chapitre 2), nous calculons en premier le degré de satisfaction des formules ϕ_1, ϕ_2, ϕ_3 et ϕ_4 pour toute interprétation.
Par exemple, soit $I = \{$*Air-France*, *Formule-Hotel*$\}$, l'utilisation de la définition 2.3 (relation d'inférence des formules de choix de base) pour les préférences représentées par ϕ_1 et ϕ_4, nous donne :

- $I \models \neg$*EasyJet* $\vee$ $\neg$*Air-France*, donc $I \models \phi_1$, et
- $I \models_1$ *Air-France* $\vec{\times}$ *EasyJet*, donc $I \models_1 \phi_4$.

Considérons maintenant la préférence ϕ_2, c-à-d. $\neg$(*Air-France* $\vec{\times}$ *EasyJet*) $\vee$ *Formule-Hotel*. Nous avons $I \not\models \neg$ (*Air-France* $\vec{\times}$ *EasyJet*) et $I \models$ *Formule-Hotel*. Par l'application de l'item (3) de la définition 2.5, nous obtenons $I \models_1 \neg$ (*Air-France* $\vec{\times}$ *EasyJet*) $\vee$ *Formule-Hotel*. Concernant

la préférence ϕ_3, nous avons, $I \not\models \neg(\textit{EasyJet} \vec{\times} \textit{Air-France}) \vee \neg \textit{Formule-Hotel}$.

L'interprétation I satisfait les formules ϕ_1, ϕ_2 et ϕ_4, mais falsifie ϕ_3, donc I n'est pas un modèle pour la base $K \cup T$, il ne peut donc pas être un modèle préféré.

Le tableau suivant montre si chaque formule ϕ_1, ϕ_2, ϕ_3 et ϕ_4 de K et T est satisfaite (T) ou non (-) pour toute interprétation.

Air-France	*EasyJet*	*Formule-Hotel*	ϕ_1	ϕ_2	ϕ_3	ϕ_4
F	F	F	T	T	T	-
F	F	T	T	T	T	-
F	T	F	T	-	T	T
F	T	T	T	T	-	T
T	F	F	T	-	T	T
T	F	T	T	T	-	T
T	T	F	-	-	T	T
T	T	T	-	T	-	T

TAB. 3.1: Les modèles de la base $K \cup T$ dans le cadre de la relation d'inférence $\mathcal{QCL}$

Nous constatons qu'il n'existe aucune interprétation qui satisfait, à la fois, toutes les formules ϕ_1, ϕ_2, ϕ_3 et ϕ_4, d'où la base $K \cup T$ n'a aucun modèle préféré. Donc, nous avons $K \cup T \mid\sim^{QCL}$ *Formule-Hotel* et $K \cup T \mid\sim^{QCL} \neg$ *Formule-Hotel*, alors que seulement $K \cup T \not\mid\sim^{QCL}$ *Formule-Hotel* est désirable.

3.2.3 La double négation

La double négation d'une formule $\mathcal{QCL}$ donne une formule propositionnelle non équivalente à la formule originale :

$$\forall\ \phi \in GCF_{PS},\ \neg\neg(\phi) \neq \phi.$$

Pour illustrer que $\neg\neg(\phi) \neq \phi$, nous supposons que la formule ϕ est de la forme $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, où les a_i sont des formules propositionnelles, i = 1,...,n. Par l'application de la négation d'une formule donnée dans l'équation (3.1), nous avons :

$$\begin{aligned}\neg\neg(\phi) &\equiv \neg\neg(a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n)\\ &\equiv \neg(\neg a_1 \wedge \neg a_2 \wedge ... \wedge \neg a_n)\\ &\equiv \neg\neg a_1 \vee \neg\neg a_2 \vee ... \vee \neg\neg a_n\\ &\equiv a_1 \vee a_2 \vee ... \vee a_n\end{aligned}$$

D'aprés la définition 2.6, les deux formules ϕ et $\neg\neg\phi$ sont dites équivalentes ssi :

1. $\text{opt}(\phi) = \text{opt}(\neg\neg\phi)$,
2. et pour toute interprétation I, et entier k, nous avons $I \mid\sim_k^{\mathcal{QCL}} \phi$ ssi $I \mid\sim_k^{\mathcal{QCL}} \neg\neg\phi$.

$$\begin{aligned}
\text{Or, } \mathrm{opt}(\phi) &= \mathrm{opt}(a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n) \\
&= \mathrm{opt}(a_1) + \mathrm{opt}(a_2) + \ldots + \mathrm{opt}(a_n) \\
&= 1 + 1 + ... + 1 \text{ (car les } a_i \text{ sont des formules propositionnelles)} \\
&= \mathrm{n}. \\
\text{Et } \mathrm{opt}(\neg\neg\phi) &= \mathrm{opt}(\neg\neg(a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n)) \\
&= \mathrm{opt}\ (\neg(\neg a_1 \wedge \neg a_2 \wedge ... \wedge \neg a_n)) \\
&= \mathrm{opt}(a_1 \vee a_2 \vee ... \vee a_n) \\
&= 1.
\end{aligned}$$

Ce qui montre en général que $\mathrm{opt}(\neg\neg\phi) \neq \mathrm{opt}(\phi)$.

D'une façon similaire, étant donnée une interprétation I, nous montrons si $\neg\neg\phi$ et ϕ ont le même degré de satisfaction :

D'un coté, nous avons $I \mid\!\sim_k^{\mathcal{QCL}} \phi$ ssi $I \mid\!\sim_k^{\mathcal{QCL}} a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$. D'un autre coté, $I \mid\!\sim_k^{\mathcal{QCL}} \neg\neg\phi$ ssi $I \mid\!\sim_k^{\mathcal{QCL}} a_1 \vee a_2 \vee ... \vee a_n$.
Si $I \mid\!\sim_k^{\mathcal{QCL}} a_1 \vee a_2 \vee ... \vee a_n$ alors $k = 1$ (si la formule $a_1 \vee a_2 \vee ... \vee a_n$ est satisfaite, son degré de satisfaction est unique et toujours égal à 1), alors que la formule $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ peut être satisfaite avec le degré $1 \leq k \leq n$.
Si $I \not\mid\!\sim^{\mathcal{QCL}} a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_{n-1}$ et $I \mid\!\sim_1^{\mathcal{QCL}} a_n$ alors $I \mid\!\sim_k^{\mathcal{QCL}} \phi$ avec $k = n$, et $I \mid\!\sim_1^{\mathcal{QCL}} \neg\neg\phi$.

L'optionalité des deux formules ϕ et $\neg\neg\phi$ est différente, et leurs degrés de satisfaction sont aussi différents, ce qui montre en général que $\neg\neg\phi \neq \phi$.

Nous pouvons conclure donc que la relation d'inférence $\mathcal{QCL}$ est satisfaisante lorsqu'un agent exprime ses préférences uniquement par des formules de choix de base. Cependant, lorsque ces préférences sont représentées par des formules de choix générales, notamment en présence de la négation ou d'implication, la relation d'inférence est totalement insatisfaisante.

3.3 Révisions et modifications de la logique $\mathcal{QCL}$

Comme nous l'avons constaté dans la section précédente, la présence de la négation et des conditionnels dans les préférences exprimées pose une vraie limite de la logique $\mathcal{QCL}$. Pour résoudre les problèmes posés par ces limites, il est indispensable de modifier la négation définie dans le cadre $\mathcal{QCL}$. Cette première modification donne ainsi une nouvelle logique que nous avons appelée *Logique du Choix Qualitatif Minimale ($\mathcal{MQCL}$)*, qui se caractérise par une nouvelle définition de la négation, mais garde les définitions de conjonction et de disjonction des préférences de la logique $\mathcal{QCL}$. La logique $\mathcal{MQCL}$ est intéressante, car elle garde la définition de la conjonction de $\mathcal{QCL}$ (voir item 4-(a) de la définition 3.1) qui est assez intéressante et propose une nouvelle négation qui permet de surmonter les limites de $\mathcal{QCL}$. Par ailleurs, se limiter uniquement à modifier la définition de la négation n'est pas intéressant du fait, que la nouvelle logique ne vérifie pas les propriétés de De Morgan comme c'est le cas pour $\mathcal{QCL}$ et ne permet pas de représenter d'autres types de préférences telles que les préférences prioritaires et positives. C'est pourquoi, nous proposons d'apporter d'autres modifications à la logique $\mathcal{QCL}$ au niveau de la conjonction et de la disjonction des préférences. Ces modifications donnent deux autres logiques : la première est appelée *Logique du Choix Qualitatif Prioritaire*, notée par $\mathcal{PQCL}$, et la seconde est appelée *Logique du Choix Qualitatif Positive*, notée par $\mathcal{QCL}+$. Ces deux logiques se caractérisent par une nouvelle définition de la négation qui est, également, la même que $\mathcal{MQCL}$

et des nouvelles définitions de la conjonction et de la disjonction. Dans le cadre de la logique $\mathcal{PQCL}$, la conjonction (resp. disjonction) des préférences est prioritaire, dans le cadre de $\mathcal{QCL}+$, la conjonction (resp. disjonction) des préférences est positive.

Ainsi, les révisions et modifications que nous avons apportées à la logique $\mathcal{QCL}$ donnent le résultat de trois logiques qui permettent de surmonter les limites du cadre $\mathcal{QCL}$ original. Les modifications apportées à $\mathcal{QCL}$ sont résumées dans le schéma de la figure 3.1 :

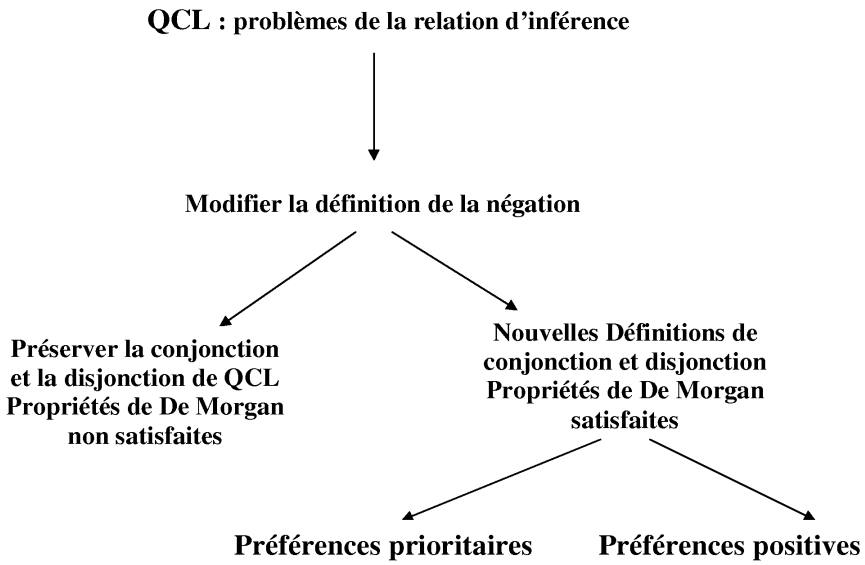

FIG. 3.1: Modification de $\mathcal{QCL}$ et propositions

Chacune des logiques proposées se caractérise par une sémantique bien particulière, en ce qui concerne le type des préférences qu'elle permet de représenter, mais les trois, partagent plusieurs propriétés comme la définition de la négation des préférences par exemple. Les trois logiques $\mathcal{MQCL}$, $\mathcal{PQCL}$ et $\mathcal{QCL}+$ se basent sur le même langage $\mathcal{QCL}$ (même vocabulaire et connecteurs). La différence principale et essentielle, concerne la relation d'inférence des formules de GCF_{PS} qui permet de tenir en compte, de la négation et des conditionnels.

3.3.1 La nouvelle négation dans le cadre des logiques proposées

Dans le cadre de la logique $\mathcal{QCL}$, la négation d'une préférence de la forme *"il n'est pas vrai que je préfère aller au cinéma à ne pas y aller"* est codée par *"¬(cinéma* $\vec{\times}\neg$ *cinéma)"* qui est équivalente à $\perp$ (théorie contradictoire). Lorsque nous appliquons la définition de la négation que nous avons proposée dans le cadre de nos logiques, nous n'obtenons jamais une contradiction, et nous présentons cela simplement par *"je ne préfère pas aller au cinéma plutôt que l'inverse"* codée par *"¬cinéma* $\vec{\times}$ *cinéma"*

Dans le cadre des logiques $\mathcal{MQCL}$, $\mathcal{PQCL}$ et $\mathcal{QCL}+$, nous avons donné une nouvelle définition de la négation. Dans notre définition, si nous avons des préférences représentées par $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, nous exigeons que a_1 ne doit pas être considérée comme une première option,

a_2 ne doit pas être considérée comme une deuxième option, etc. Notre définition de la négation est de la forme :

$$\neg(a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n) \equiv \neg a_1 \vec{\times} \neg a_2 \vec{\times} \ldots \vec{\times} \neg a_n \tag{3.2}$$

Dans le cadre de la logique $\mathcal{QCL}$, la négation d'une formule de choix de base ou de choix générale est équivalente à une formule propositionnelle, où d'une part, la disjonction ordonnée est remplacée par la disjonction classique, et d'autre part le degré de satisfaction de la formule obtenue (si elle est satisfaite) est toujours égal à 1. Ce résultat implique qu'il n'est plus question d'ordonner les alternatives entre elles. Par contre, dans le cadre de nos logiques ($\mathcal{MQCL}$, $\mathcal{PQCL}$ et $\mathcal{QCL}+$), la négation d'une formule garde la disjonction ordonnée et son degré de satisfaction est défini comme n'importe quelle formule de choix de base ou de choix générale.

Nous verrons dans la section 4.5.3 qu'il est possible de donner une autre définition de la négation qui est utilisée dans la théorie des possibilités.

3.4 $\mathcal{MQCL}$: Logique du Choix Qualitatif Minimale

La logique du choix qualitatif minimale $\mathcal{MQCL}$ ressemble à $\mathcal{QCL}$ du fait qu'elle est également une extension de la logique propositionnelle. Plus précisément, lorsque les bases de connaissances ne contiennent que des formules propositionnelles, la relation d'inférence $\mathcal{MQCL}$ se comporte comme l'inférence propositionnelle. En outre, lorsque les agents expriment leurs préférences uniquement par des formules de choix de base, l'inférence $\mathcal{MQCL}$ est totalement équivalente à la relation d'inférence $\mathcal{QCL}$. Cependant, lorsque nous traitons des préférences représentées par des formules de choix générales, la relation d'inférence $\mathcal{MQCL}$ se montre plus satisfaisante que l'inférence $\mathcal{QCL}$. En effet, la relation d'inférence de la logique $\mathcal{MQCL}$ se caractérise par une fonction de normalisation. Elle propose une nouvelle définition de la négation permettant de surmonter les limites de la logique $\mathcal{QCL}$ et garde les mêmes définitions de la conjonction et de la disjonction de $\mathcal{QCL}$. Nous présentons la fonction normalisation de $\mathcal{MQCL}$ dans la section suivante.

3.4.1 Fonction de normalisation de la logique $\mathcal{MQCL}$

Dans cette section, nous montrons que tout ensemble de préférences représentées par des formules de choix générales peut être transformé d'une manière équivalente en un ensemble de préférences en formes normales représentées par des formules de choix de base. Cette transformation est obtenue par l'utilisation de ce que l'on appelle *fonction de normalisation*, notée par $\mathcal{N}_{\mathcal{MQCL}}$. Comme nous l'avons déjà dit dans l'introduction de ce chapitre, cette transformation s'avère intéressante car elle permet de présenter les préférences en formes normales. $\mathcal{N}_{\mathcal{MQCL}}$ est définie par :

Définition 3.1
Une fonction de normalisation, notée par $\mathcal{N}_{\mathcal{MQCL}} : GCF_{PS} \rightarrow BCF_{PS}$, est une fonction qui transforme toute formule de choix générale en une formule de choix de base, tel que :

(1) La forme normale de toute formule de choix de base ou formule propositionnelle correspond à la formule elle même : $\forall\ \phi \in BCF_{PS}, \mathcal{N}_{\mathcal{MQCL}}(\phi) = \phi$.

(2) La forme normale de la négation des formules est donnée comme suit :
$\forall\ \phi \in GCF_{PS}$, *et* $(\phi \notin BCF_{PS})$, $\mathcal{N}_{\mathcal{MQCL}}(\neg\phi) \equiv \mathcal{N}_{\mathcal{MQCL}}(\neg\mathcal{N}_{\mathcal{MQCL}}(\phi))$.

(3) La forme normale est décomposable respectivement par rapport à la conjonction, disjonction et disjonction ordonnée des formules de choix générales :

(a) $\forall\ \phi, \psi \in GCF_{PS}$ *et* $(\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS})$,
$\mathcal{N}_{\mathcal{MQCL}}(\phi \wedge \psi) \equiv \mathcal{N}_{\mathcal{MQCL}}(\mathcal{N}_{\mathcal{MQCL}}(\phi) \wedge \mathcal{N}_{\mathcal{MQCL}}(\psi))$.

(b) $\forall\ \phi, \psi \in GCF_{PS}$ *et* $(\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS})$,
$\mathcal{N}_{\mathcal{MQCL}}(\phi \vee \psi) \equiv \mathcal{N}_{\mathcal{MQCL}}(\mathcal{N}_{\mathcal{MQCL}}(\phi) \vee \mathcal{N}_{\mathcal{MQCL}}(\psi))$.

(c) $\forall\ \phi, \psi \in GCF_{PS}$ *et* $(\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS})$,
$\mathcal{N}_{\mathcal{MQCL}}(\phi \vec{\times} \psi) \equiv \mathcal{N}_{\mathcal{MQCL}}(\mathcal{N}_{\mathcal{MQCL}}(\phi) \vec{\times} \mathcal{N}_{\mathcal{MQCL}}(\psi))$.

(4) La normalisation de la conjonction, disjonction et négation des formules de choix de base est donnée comme suit :

Soient $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, *et* $\psi = b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m$ *telles que* $a_{i=1,\ldots n}$ *et* $b_{j=1,\ldots m}$ *sont des formules propositionnelles,*

(a) $\mathcal{N}_{\mathcal{MQCL}}((a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n) \wedge (b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m)) \equiv c_1 \vec{\times} c_2 \vec{\times} \ldots \vec{\times} c_k$, *où* $k = max(m, n)$, *et*

$$c_i = \begin{cases} [(a_1 \vee \ldots \vee a_i) \wedge b_i] \vee [a_i \wedge (b_1 \vee \ldots \vee b_i)] \\ \quad si \quad i \leq min(m,n) \\ ((a_1 \vee \ldots \vee a_n) \wedge b_i) \quad si \quad n < i \leq m \\ (a_i \wedge (b_1 \vee \ldots \vee b_m)) \quad si \quad m < i \leq n \end{cases}$$

(b) $\mathcal{N}_{\mathcal{MQCL}}(\phi \vee \psi) \equiv c_1 \vec{\times} c_2 \vec{\times} \ldots \vec{\times} c_k$ *où* $k = max(m, n)$, *et*

$$c_i = \begin{cases} (a_i \vee b_i) & si \quad i \leq min(m,n) \\ a_i & si \quad m \leq i \leq n \\ b_i & si \quad n \leq i \leq m \end{cases}$$

(c) $\mathcal{N}_{\mathcal{MQCL}}\ \neg(\ a_1 \vec{\times}\ a_2 \vec{\times} \ldots \vec{\times}\ a_n) \equiv \neg\ a_1 \vec{\times} \neg\ a_2 \vec{\times} \ldots \vec{\times} \neg\ a_n$.

La propriété (1) indique que la forme normale d'une formule de choix de base ϕ, donne la formule ϕ. La propriété (2) donne la normalisation de la négation d'une formule de choix générale. La propriété 3 (a, b, c) indique que la fonction de normalisation est décomposable par rapport à la conjonction, disjonction et la disjonction ordonnée. La propriété 4 (a, b, c) donne les définitions de la conjonction, disjonction et la négation des formules de choix de bases.
Concernant la conjonction de deux formules de choix de base (propriété 4-(a)), supposons que $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ représente les préférences d'un agent A, et $b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m$ représente les préférences de l'agent B. L'application de la conjonction des préférences permet de sélectionner des solutions qui donnent le même privilège à A et B. Par exemple, $a_1 b_1$ (qui représente le meilleur choix pour A et le meilleur choix pour B) est préféré à ($a_1 b_2$ ou $a_2 b_2$ ou $a_2 b_1$) (qui représente soit le premier choix pour A et le second choix pour B, soit le second choix pour A et B, soit le second choix pour A et le premier choix pour B). Cette conjonction est assez forte du fait que,

l'ordre des meilleures alternatives est plus exigeant que les autres alternatives moins préférées. Dans cet exemple, nous avons uniquement une possibilité de satisfaire la meilleure solution qui consiste à choisir l'option a_1b_1, alors que pour la seconde solution, trois options sont possibles qui sont a_1b_2, a_2b_2 et a_2b_1.

3.4.2 La relation d'inférence $\mathcal{MQCL}$

La sémantique de toute formule est basée sur son degré de satisfaction dans un modèle particulier I. Si une interprétation I satisfait une formule ϕ, alors son degré de satisfaction doit être unique. Nous utiliserons la notation $I|\sim_i^{MQCL}\phi$ pour exprimer que I satisfait ϕ avec le degré i.

Dans le cadre de cette logique, la relation d'inférence des formules propositionnelles et formules de choix de base, est la même que celle définie pour la logique $\mathcal{QCL}$ (Définition 2.3). Concernant les formules de choix générales, la relation d'inférence est donnée comme suit :

- Appliquer la définition 3.1 pour transformer toute formule de choix générale en une formule de choix de base.
- Appliquer la définition 2.3 pour toutes les formules de choix de base obtenues.

Exemple 3.2
Soit la formule de choix générale suivante : $\psi = (a_1 \vec{\times}\ a_2) \wedge (b_1 \vec{\times}\ b_2)$.
Pour donner le degré de satisfaction de la formule ψ pour toute interprétation, nous normalisons d'abord ψ (c-à-d. nous transformons ψ en une formule de choix de base équivalente) comme c'est indiqué dans la définition 3.1. Par la suite, nous utilisons la définition 2.3 pour donner la relation d'inférence de la formule de choix de base obtenue.

1. Normalisation de ψ : nous utilisons l'item 4-(a) de la définition 3.1, nous obtenons :
   $\mathcal{N}_{\mathcal{MQCL}}(\psi)$=$\mathcal{N}_{\mathcal{MQCL}}$ $((a_1 \vec{\times} a_2) \wedge (b_1 \vec{\times} b_2))$
   $\equiv ((a_1 \wedge b_1) \vee (a_1 \wedge b_1)) \vec{\times}(((a_1 \vee a_2) \wedge b_2) \vee (a_2 \wedge (b_1 \vee b_2)))$.
   $\equiv (a_1 \wedge b_1) \vec{\times}\ ((a_1 \wedge b_2) \vee (a_2 \wedge b_2) \vee (a_2 \wedge b_1))$.

2. Degré de satisfaction de ψ :
   - Soit $I_1 = \{a_2, b_2\}$. L'application de l'item 2 de la définition 2.3, donne :
     $I_1 \not\models a_1 \wedge b_1$, $I_1 \not\models a_1 \wedge b_2$, $I_1 \not\models a_2 \wedge b_1$, et $I_1 \models a_2 \wedge b_2$, donc
     $I_1 \models (a_1 \wedge b_2) \vee (a_2 \wedge b_2) \vee (a_2 \wedge b_1)$.
     L'utilisation de l'item 1 de la définition 2.3 donne : $I_1 \models_2 \mathcal{N}_{\mathcal{MQCL}}(\psi)$. Donc I_1 satisfait ψ avec le degré 2.
   - Si $I_2 = \{b_2\}$ alors $I_2 \not\models \mathcal{N}_{\mathcal{MQCL}}(\psi)$. I_2 ne satisfait pas ψ.
   - Si $I_3 = \{a_1, b_1\}$ alors $I_3 \models_1 \mathcal{N}_{\mathcal{MQCL}}(\psi)$. I_3 satisfait ψ avec le degré 1.
     C'est-à-dire, ψ est totalement satisfaite par I_3.

3.4.3 L'inférence à partir d'un ensemble de connaissances et préférences

Soient K un ensemble de formules propositionnelles permettant de représenter les connaissances de l'agent ou les contraintes d'intégrité, et T soit un ensemble de formules de choix de base ou de choix générales pour représenter les préférences. La définition des modèles préférés dans le cadre de la logique $\mathcal{MQCL}$ est similaire à celle donnée dans [Brewka *et al.* 2004] (Définition 2.8), mais nous utilisons la relation d'inférence $\mathcal{MQCL}$ pour définir le degré de satisfaction de chaque formule.

Nous définissons la relation d'inférence entre les formules de la base $K \cup T$ et une formule propositionnelle ψ comme suit :

Définition 3.2
Soient K un ensemble de formules propositionnelles et T un ensemble de préférences représentées par des formules de choix générales. Soit T' un ensemble de préférences obtenues de T par le remplacement de chaque ϕ dans T par $\mathcal{N}_{\mathcal{MQCL}}(\phi)$.
Soit ψ une formule propositionnelle de K.
$K \cup T \mid\sim^{MQCL} \psi$ si ψ est satisfaite dans tous les modèles préférés de $K \cup T'$.

Dans ce qui suit, nous reprenons l'exemple 3.1, où la relation d'inférence $\mathcal{QCL}$ n'est pas satisfaisante, mais nous appliquons la relation d'inférence $\mathcal{MQCL}$:

Exemple 3.3
Considérons l'exemple 3.1 où la base de connaissances K contient :

$$\phi_1 = \neg \textit{EasyJet} \vee \neg \textit{Air-France}$$

et la base de préférences T contient les préférences suivantes :

$$\begin{cases} \phi_2 = \neg(Air\text{-}France \vec{\times} EasyJet) \vee Formule\text{-}Hotel \\ \phi_3 = \neg(EasyJet \vec{\times} Air\text{-}France) \vee \neg Formule\text{-}Hotel \\ \phi_4 = Air\text{-}France \vec{\times} EasyJet \end{cases}$$

Pour calculer les modèles préférés de la base $K \cup T$, nous procédons en premier à calculer le degré de satisfaction de toutes les formules ϕ_1, ϕ_2, ϕ_3 et ϕ_4 pour toute interprétation.

Concernant l'inférence des formules ϕ_1 et ϕ_4, il suffit d'appliquer la définition 2.3 (item 1 pour ϕ_4 et item 2 pour ϕ_1). Pour les formules ϕ_2 et ϕ_3, nous appliquons d'abord la définition 3.1 pour transformer ces deux formules en formules de choix de base équivalentes. Par la suite, nous appliquons la définition 2.3 pour définir les degrés de satisfaction des formules obtenues.

Par l'application des items 4-(c) et 4-(b) de la définition 3.1, nous avons :
la préférence ϕ_2 (c-à-d. $\neg$ (*Air-France* $\vec{\times}$ *EasyJet*) $\vee$ *Formule-Hotel*) est équivalente à la formule ($\neg$*Air-France* $\vee$ *Formule-Hotel*) $\vec{\times}$ $\neg$ *EasyJet*, et la préférence ϕ_3 (c-à-d. $\neg$ (*EasyJet*$\vec{\times}$ *Air-France*) $\vee$ $\neg$*Formule-Hotel*) est équivalente à la formule ($\neg$*EasyJet* $\vee$ $\neg$*Formule-Hotel*) $\vec{\times}$ $\neg$ *Air-France*.
La nouvelle base de préférences T' contient donc les préférences suivantes représentées par des formules de choix de base uniquement :

$$\begin{cases} \phi'_2 = (\neg Air\text{-}France \vee Formule\text{-}Hotel) \vec{\times} \neg EasyJet \\ \phi'_3 = (\neg EasyJet \vee \neg Formule\text{-}Hotel) \vec{\times} \neg Air\text{-}France \\ \phi_4 = Air\text{-}France \vec{\times} EasyJet \end{cases}$$

Par exemple, soit $I = \{$*Air-France*, *Formule-Hotel*$\}$:
Nous avons $I \mid\sim_1^{MQCL} (\neg$*Air-France* $\vee$ *Formule-Hotel*) $\vec{\times}$ $\neg$ *EasyJet*, donc le degré de satisfaction de la formule ϕ'_2 est égal à 1, étant donnée l'interprétation I.

Le tableau suivant donne le degré de satisfaction de chaque formule ϕ_1, ϕ'_2, ϕ'_3 et ϕ_4 de K et T'. Pour toute interprétation, le nombre (1, 2) indique le degré de satisfaction de la formule si elle satisfaite, ou (-) si elle est falsifiée.

Air-France	*EasyJet*	*Formule-Hotel*	ϕ_1	$\phi_{'2}$	$\phi_{'3}$	ϕ_4
F	F	F	1	1	1	-
F	F	T	1	1	1	-
F	T	F	1	1	1	2
F	T	T	1	1	2	2
T	F	F	1	2	1	1
T	**F**	**T**	**1**	**1**	**1**	**1**
T	T	F	-	-	1	1
T	T	T	-	1	-	1

TAB. 3.2: Les modèles de la base $K \cup T'$ dans le cadre de la relation d'inférence $\mathcal{MQCL}$

Dans le cadre de la logique $\mathcal{QCL}$, $K \cup T$ est incohérente (comme nous l'avons montré dans l'exemple 3.1). Avec la logique $\mathcal{MQCL}$, $K \cup T$ admet un seul modèle préféré (ligne en gras), $I = \{$*Air-France*, *Formule-Hotel*$\}$, d'où le résultat $K \cup T \mid\sim^{MQCL}$ *Formule-Hotel*, qui correspond au résultat attendu que l'agence doit proposer à ces clients lorsqu'elle prend en considération leurs préférences.

3.4.4 Caractéristiques de la logique $\mathcal{MQCL}$

Comme nous l'avons déjà souligné, la logique $\mathcal{MQCL}$ est une simple extension de $\mathcal{QCL}$ puisque la conjonction (resp. disjonction) des préférences est définie d'une façon similaire. Par ailleurs, comme le principal problème de $\mathcal{QCL}$ réside dans l'utilisation de la négation dans les préférences, la logique $\mathcal{MQCL}$ définit différemment la négation dans le but de surmonter les limites de $\mathcal{QCL}$. Les principales caractéristiques de $\mathcal{MQCL}$ sont les suivantes :

- La relation d'inférence de $\mathcal{MQCL}$ est satisfaisante, c'est-à-dire que la présence de la négation dans les préférences ne pose aucun problème.
- L'optionalité de la négation d'une formule ϕ est égale à l'optionalité de ϕ (dans le cadre de la logique $\mathcal{QCL}$, l'optionalité de la négation d'une formule ϕ est égale à 1). Nous donnons la définition de l'optionalité dans le cadre $\mathcal{MQCL}$ dans ce qui suit (notons que cette définition est une version révisée de celle donnée dans [Brewka *et al.* 2004] que nous avons également rappelée dans le chapitre précédent (Définition 2.4)) :

  Définition 3.3
  Soient ϕ_1 et ϕ_2 deux formules dans GCF_{PS}.
  – *opt(a) = 1, a est un atome propositionnel.*
  – *opt($\phi_1 \vec{\times} \phi_2$) = opt($\phi_1$)+ opt($\phi_2$).*
  – *opt($\phi_1 \wedge \phi_2$) = max(opt(ϕ_1), opt(ϕ_2)).*
  – *opt($\phi_1 \vee \phi_2$) = max(opt(ϕ_1), opt(ϕ_2)).*
  – *opt($\neg(\phi_1)$) = opt(ϕ_1).*

  L'optionalité d'une formule indique le plus grand nombre (entier positif) avec lequel une formule peut être satisfaite dans le pire des cas. Si $\phi = a_1 \vec{\times} a_2 \vec{\times} ... \vec{\times} a_n$, alors elle est satisfaite dans le pire des cas lorsqu'il existe uniquement une interprétation qui la satisfait au degré n (uniquement l'option a_n qui est satisfaite), l'optionalité de cette formule dans ce cas

est égale à n. L'optionalité de la conjonction (resp. disjonction) de deux formules est le maximum des options que chaque formule présente. Donc, la définition de l'optionalité de la conjonction et la disjonction dans $\mathcal{MQCL}$ est la même de celle utilisée dans la logique $\mathcal{QCL}$. Par ailleurs, la définition de l'optionalité de la négation, est différente à celle définie dans la logique $\mathcal{QCL}$.

- La double négation d'une formule $\mathcal{MQCL}$ (BCF ou GCF) donne la même formule.
- Les propriétés de De Morgan ne sont pas vérifiées, c'est-à-dire que nous avons :
  - $\mathcal{N}_{\mathcal{MQCL}}(\neg(\phi_1 \wedge \phi_2)) \neq \mathcal{N}_{\mathcal{MQCL}}(\neg\phi_1 \vee \neg\phi_2)$, et
  - $\mathcal{N}_{\mathcal{MQCL}}(\neg(\phi_1 \vee \phi_2)) \neq \mathcal{N}_{\mathcal{MQCL}}(\neg\phi_1 \wedge \neg\phi_2)$.

  Par contre, nous avons :

  - $\mathcal{N}_{\mathcal{MQCL}}(\neg(\phi_1 \wedge \phi_2)) \equiv \mathcal{N}_{\mathcal{MQCL}}(\neg\mathcal{N}_{\mathcal{MQCL}}(\phi_1 \wedge \phi_2))$, et
  - $\mathcal{N}_{\mathcal{MQCL}}(\neg(\phi_1 \vee \phi_2)) \equiv \mathcal{N}_{\mathcal{MQCL}}(\neg\mathcal{N}_{\mathcal{MQCL}}(\phi_1 \vee \phi_2))$.

Exemple 3.4
Nous illustrons dans cet exemple $\mathcal{N}_{\mathcal{MQCL}}(\neg(\phi_1 \wedge \phi_2)) \neq \mathcal{N}_{\mathcal{MQCL}}(\neg\phi_1 \vee \neg\phi_2)$.

Soit la formule de choix générale suivante : $\psi = (a_1 \vec{\times}\ a_2) \wedge\ (b_1 \vec{\times}\ b_2)$. Nous appliquons respectivement l'item (2), l'item 4-(a) et l'item 4-(c) de la définition 3.1. Nous avons :

$\mathcal{N}_{\mathcal{MQCL}}(\neg\psi) = \mathcal{N}_{\mathcal{MQCL}}(\neg((a_1 \vec{\times}\ a_2) \wedge\ (b_1 \vec{\times}\ b_2)))$
$\equiv \mathcal{N}_{\mathcal{MQCL}}(\neg\mathcal{N}_{\mathcal{MQCL}}((a_1 \vec{\times} a_2) \wedge (b_1 \vec{\times} b_2)))$
$\equiv \mathcal{N}_{\mathcal{MQCL}}(\neg\ (((a_1 \wedge b_1) \vee (a_1 \wedge b_1))\ \vec{\times}((a_1 \vee a_2) \wedge b_2) \vee (a_2 \wedge (b_1 \vee b_2)))).$
$\equiv \mathcal{N}_{\mathcal{MQCL}}(\neg\ ((a_1 \wedge b_1) \vec{\times}\ ((a_1 \wedge b_2) \vee (a_2 \wedge b_2) \vee (a_2 \wedge b_1)))).$
$\equiv \mathcal{N}_{\mathcal{MQCL}}(\neg(a_1 \wedge b_1)\ \vec{\times}\ \neg((a_1 \wedge b_2) \vee (a_2 \wedge b_2) \vee (a_2 \wedge b_1))).$
$\equiv \neg(a_1 \wedge b_1)\ \vec{\times}\ \neg(a_1 \wedge b_2) \wedge \neg(a_2 \wedge b_2) \wedge \neg(a_2 \wedge b_1).$

D'un autre coté, par l'application des items 3-(b), 4-(c) et 4-(b) de la définition 3.1, nous avons :

$\mathcal{N}_{\mathcal{MQCL}}(\neg(a_1 \vec{\times}\ a_2) \vee\ \neg(b_1 \vec{\times}\ b_2))$
$\equiv = \mathcal{N}_{\mathcal{MQCL}}\ (\mathcal{N}_{\mathcal{MQCL}}(\neg(a_1 \vec{\times}\ a_2)) \vee\ \mathcal{N}_{\mathcal{MQCL}}(\neg(b_1 \vec{\times}\ b_2)))$
$\equiv \mathcal{N}_{\mathcal{MQCL}}((\neg a_1 \vec{\times} \neg a_2)\ \vee(\neg b_1 \vec{\times} \neg b_2))$
$\equiv \mathcal{N}_{\mathcal{MQCL}}((\neg a_1 \vee \neg b_1) \vec{\times} (\neg a_2 \vee \neg b_2))$
$\equiv (\neg a_1 \vee \neg b_1) \vec{\times} (\neg a_2 \vee \neg b_2)$

Les résultats de $\mathcal{N}_{\mathcal{MQCL}}\ (\neg((a_1 \vec{\times}\ a_2) \wedge\ (b_1 \vec{\times}\ b_2))$ et $\mathcal{N}_{\mathcal{MQCL}}\ (\neg(a_1 \vec{\times}\ a_2)) \vee\ \neg(b_1 \vec{\times}\ b_2))$ sont différents, ce qui donc confirme que $\mathcal{N}_{\mathcal{MQCL}}(\neg(\phi_1 \wedge \phi_2)) \neq \mathcal{N}_{\mathcal{MQCL}}(\neg\phi_1 \vee \neg\phi_2)$.

3.5 Conclusion

Dans ce chapitre, nous avons présenté les limites de la logique $\mathcal{QCL}$, qui se concentrent au niveau du raisonnement sur les préférences conditionnelles et en présence de la négation. Pour remédier aux problèmes posés par ces limite, nous avons également présenté les révisions et les modifications apportées à cette logique. Ces modifications ont concerné principalement des

nouvelles définitions de la négation, de la conjonction et de la disjonction des préférences. La nouvelle négation permet de surmonter les limites de la logique $\mathcal{QCL}$ d'une part, et de raisonner sur les préférences conditionnelles, d'autre part. Quant aux nouvelles définitions de la conjonction et de la disjonction, de plus qu'elles permettent de vérifier certaines propriétés importantes dans le cadre logique (propriétés de De Morgan et double négation d'une formule), elles permettent de représenter d'autres types de préférences (prioritaires et positives). Dans ce chapitre, nous avons présenté la première logique $\mathcal{MQCL}$, qui est une simple extension de $\mathcal{QCL}$. Elle se caractérise par une nouvelle définition de la négation et les mêmes définitions de la conjonction et de la disjonction de $\mathcal{QCL}$. Les deux autres logiques de représentation des préférences prioritaires et positives seront présentées dans le chapitre suivant.

Chapitre 4

Représentation des préférences prioritaires et positives

Sommaire

4.1 Introduction

Lorsque les agents effectuent leurs choix sur un ensemble d'alternatives, ils se basent généralement sur ce qu'ils souhaitent fortement, partiellement ou faiblement, ce qu'ils aimeraient et ce qu'ils ne souhaitent pas ou n'aiment pas. Cela signifie que différents critères interviennent dans leurs choix tels que les priorités, les buts, les rejets, etc. C'est pourquoi, il existe plusieurs types des préférences telles que les préférences positives, négatives [Bistarelli *et al.* 2005], conditionnelles [Wilson 2004, Boutilier 1994], etc.

Dans ce chapitre, nous nous intéressons aux préférences prioritaires et positives. Ces préférences sont représentées dans le cadre de la logique $\mathcal{QCL}$ à laquelle nous avons apportée des

modifications au niveau des définitions de la négation, de la conjonction et de la disjonction des préférences. Comme c'est indiqué dans le chapitre précédent, les modifications et les révisions de $\mathcal{QCL}$ donnent trois alternatives : $\mathcal{MQCL}$ qui est une simple extension de $\mathcal{QCL}$, $\mathcal{PQCL}$ adaptée à la représentation des préférences prioritaires et $\mathcal{QCL}+$ adaptée aux préférences positives. Ces deux dernières logiques seront présentées en détail dans ce chapitre.

Le reste du chapitre est organisé comme suit. Dans la section 4.2, nous présentons la logique $\mathcal{PQCL}$ qui permet de représenter des préférences prioritaires. Les caractéristiques principales de la logique $\mathcal{PQCL}$ sont décrites dans la section 4.2.1, la relation d'inférence et la normalisation de $\mathcal{PQCL}$ seront détaillées respectivement dans les sections 4.2.2 et 4.2.4. La section 4.3 décrit la logique des préférences positives $\mathcal{QCL}+$. Les propriétés essentielles de nos logiques et les relations avec la logique possibiliste seront présentées respectivement dans les sections 4.4 et 4.5. La section 4.6 conclut le chapitre.

4.2 $\mathcal{PQCL}$: une logique pour le traitement des préférences prioritaires

La logique $\mathcal{PQCL}$ [Benferhat & Sedki 2007, Benferhat & Sedki 2008b] est adaptée à la représentations des préférences prioritaires, c'est-à-dire des préférences ayant différents niveaux de priorité. Notre choix pour ce type de logique est motivé par le fait que les préférences des agents sont certainement différentes au niveau de l'importance qu'elles procurent. Pour un problème donné, les agents expriment souvent plusieurs préférences sur un grand nombre d'alternatives afin de pouvoir satisfaire au minimum une des préférences souhaitées même si toutes les préférences exprimées ne sont pas totalement souhaitables. Prenons un exemple où un agent doit acheter en urgence un billet pour un voyage. Supposons que ses préférences sont exprimées comme suit :

- *Il préfère voyager en avion qu'en train.*
- *Il préfère voyager le soir plutôt que le matin.*
- *Il préfère voyager en première classe plutôt qu'en seconde classe.*
- *Il préfère un voyage sans escale.*
- *Il préfère le voyage le moins cher.*

Il est clair que même si toutes ces préférences exprimées ne peuvent pas être satisfaites, l'agent préfère quand même acheter le billet disponible vu l'urgence de la situation. Ainsi, cette logique permet de donner des priorités aux préférences exprimées. Pour ces préférences, il est par exemple plus intéressant de considérer la préférence *"Il préfère voyager en avion qu'en train"* prioritaire à la préférence *"Il préfère voyager le soir plutôt que le matin"*. C'est-à-dire que s'il y a une place disponible dans un avion ou dans un train, l'agent peut faire abstraction de la contrainte du temps (soir ou matin).

Comme $\mathcal{MQCL}$, la logique du choix qualitatif prioritaire $\mathcal{PQCL}$ ressemble aussi à $\mathcal{QCL}$ car elle est également une extension de la logique propositionnelle. Plus précisément, l'inférence $\mathcal{PQCL}$ concernant les formules propositionnelles se comporte comme l'inférence propositionnelle. En outre, l'inférence $\mathcal{PQCL}$ concernant les formules de choix de base est totalement équivalente à la relation d'inférence $\mathcal{QCL}$. Cependant, lorsque nous traitons des préférences représentées par des formules de choix générales, la relation d'inférence $\mathcal{PQCL}$ se montre plus satisfaisante que l'inférence $\mathcal{QCL}$. En effet, elle suit le même principe que l'inférence $\mathcal{QCL}$ et se caractérise par de nouvelles définitions de la négation, conjonction et disjonction qui sont utiles pour l'agrégation

des préférences des utilisateurs ayant différents niveaux de priorité.

4.2.1 Caractéristiques principales de $\mathcal{PQCL}$

Les principales caractéristiques de notre logique $\mathcal{PQCL}$ sont les suivantes :

- Le langage de $\mathcal{PQCL}$ est le même que celui de $\mathcal{QCL}$.
- Si ϕ est une formule de choix de base, alors la relation d'inférence $\mathcal{PQCL}$ est équivalente à celle de $\mathcal{QCL}$, et si ϕ est une formule propositionnelle, l'inférence $\mathcal{PQCL}$ est équivalente à l'inférence de la logique classique.

- La sémantique de toute formule est basée sur son degré de satisfaction dans un modèle particulier I. Si une interprétation I satisfait une formule ϕ, alors son degré de satisfaction doit être unique. Nous utiliserons la notation $I|\sim_i^{PQCL}\phi$ pour exprimer que I satisfait ϕ avec le degré i.

- La négation : la négation doit être aussi similaire que possible que celle de la logique propositionnelle. En particulier, la double négation d'une formule doit permettre de retrouver la formule originale. Nous voulons plus précisément avoir $\neg\neg(\phi)$ équivalente à ϕ. Notons que nous ne pouvons pas simplement définir $I|\sim_i\neg\phi$ si et seulement si "$I|\sim_i\phi$ n'est pas vrai". En effet, cela implique que le degré de satisfaction de la négation d'une formule n'est pas unique (ce qui n'est pas souhaitable). Ainsi, si nous acceptons que $I|\sim_i\neg\phi$ si et seulement si $I|\sim_i\phi$ n'est pas vrai, alors si une interprétation I satisfait ϕ avec le degré 1 (c-à-d. $I|\sim_1\phi$), alors cela signifie que $I|\sim_2\neg\phi$ et $I|\sim_3\neg\phi$ sont valides (mais $I|\sim_2\neg\phi$ et $I|\sim_3\neg\phi$ sont invalides), cela signifie que $\neg\phi$ est satisfaite avec des degrés différents, ce qui n'est pas souhaitable.
  Une autre caractéristique que le cadre $\mathcal{PQCL}$ doit satisfaire, concerne la négation qui doit être décomposable par rapport à la conjonction et la disjonction, et doit également satisfaire la loi de De Morgan.

- Préférences prioritaires : notre logique $\mathcal{PQCL}$ doit être adéquate au traitement des préférences prioritaires codées par une conjonction (resp. disjonction) prioritaire. Si un agent exprime deux préférences p et q telle que p est plus importante que q, alors notre logique $\mathcal{PQCL}$ permet de coder ce type de préférences par : $p \wedge q$.

  Exemple 4.1
  Supposons qu'un agent fournit deux préférences : *"Je préfère Air-France à KLM"*, et *"Je préfère le siège fenêtre au siège sur le couloir"*. Ces préférences peuvent être codées par la formule de choix générale : "($Air\text{-}France \vec{\times} KLM$) $\wedge$ ($fenêtre \vec{\times} couloir$)" où ($Air\text{-}France \vec{\times} KLM$) doit être considérée plus importante que ($fenêtre \vec{\times} couloir$).

- Relation d'inférence : l'un des points forts de la logique $\mathcal{PQCL}$ concerne la relation d'inférence qui peut être construite de deux manières différentes mais équivalentes. La première méthode se base sur des règles d'inférence qui définissent le degré de satisfaction de chaque formule étant donnée une interprétation. La deuxième méthode est basée sur la fonction de normalisation qui permet de transformer tout ensemble de préférences représentées par des formules de choix générales en un ensemble de préférences simplifiées par des formules de choix de base.

4.2.2 La relation d'inférence $\mathcal{PQCL}$

Avant de définir formellement la relation d'inférence $\mathcal{PQCL}$, notée $\mid\!\sim^{PQCL}$, nous avons besoin d'introduire la définition de l'optionalité d'une formule dans le cadre de la logique $\mathcal{PQCL}$. Cette définition est une version révisée de celle donnée dans [Brewka *et al.* 2004] et rappelée dans la définition 2.4.

Définition 4.1
Soient ϕ_1 et ϕ_2 deux formules dans GCF_{PS}. L'optionalité d'une formule est une fonction qui assigne à chaque formule un entier strictement positif.

- *opt(a) = 1 , a est un atome propositionnel.*
- *opt($\phi_1 \vec{\times} \phi_2$)= opt($\phi_1$)+ opt($\phi_2$).*
- *opt($\phi_1 \wedge \phi_2$)= opt(ϕ_1)× opt(ϕ_2).*
- *opt($\phi_1 \vee \phi_2$)= opt(ϕ_1)× opt(ϕ_2).*
- *opt($\neg(\phi_1)$) = opt(ϕ_1).*

La différence principale avec la définition originale donnée dans [Brewka *et al.* 2004], concerne les trois derniers points de la définition 2.4. Dans [Brewka *et al.* 2004], opt($\neg\phi$) est toujours égale à 1 ($\neg\phi$ est équivalente à une formule propositionnelle). Dans le cadre de notre logique $\mathcal{PQCL}$, $\neg\phi$ et ϕ ont la même optionalité. L'optionalité de la conjonction (resp. disjonction) est aussi différente de celle de $\mathcal{QCL}$.

Dans le cadre de la logique $\mathcal{PQCL}$, la justification de l'optionalité d'une formule est principalement liée à la définition du degré de satisfaction associé par toute interprétation. Ainsi, lorsque par exemple, nos préférences sont prioritaires, il existe opt(ϕ_1)× opt(ϕ_2) options pour satisfaire ($\phi_1 \wedge \phi_2$). Cela s'explique comme suit :
Premièrement, opt(ϕ_1) (resp. opt(ϕ_2)) signifie que ϕ_1 (resp. ϕ_2) peut être satisfaite avec le degré 1 (première option de ϕ_1), avec le degré 2 (deuxième option de ϕ_1), ..., avec le degré opt(ϕ_1) (dernière option de ϕ_1), (resp. 1, 2, ..., opt(ϕ_2)). Intuitivement, étant donnée une préférence de la forme ($\phi_1 \wedge \phi_2$), la solution qui satisfait la première option de ϕ_1 et satisfait également la première option de ϕ_2 est considérée la solution la plus préférée. La seconde solution préférée est celle qui satisfait la première option de ϕ_1, mais satisfait la seconde option de ϕ_2. Ainsi, en général, une interprétation I est préférée à une interprétation I', si l'un des deux cas suivants est satisfait :

1. I satisfait ϕ_1 avec le degré i, et I' satisfait ϕ_1 avec le degré j tel que $j > i$, ou
2. I et I' satisfont ϕ_1 avec un certain degré, mais le degré avec lequel la formule ϕ_2 est satisfaite par I est inférieur au degré avec lequel ϕ_2 est satisfaite par I'.

Il est clair qu'il existe opt(ϕ_1)× opt(ϕ_2) options pour satisfaire les préférences de la forme ($\phi_1 \wedge \phi_2$). La solution la moins préférée de toutes, est celle qui satisfait la formule ϕ_1 avec un degré égal opt(ϕ_1) (la pire des options dans ϕ_1) et satisfait la formule ϕ_2 avec un degré égal opt(ϕ_2) (la pire des options dans ϕ_2).
Étant données ces optionalités associées aux formules, nous allons donc pouvoir définir la relation d'inférence pour le cadre $\mathcal{PQCL}$ qui pour chaque formule de choix générale dans GCF_{PS} et une interprétation donnée définit un degré de satisfaction. Cette relation d'inférence est définie par :

Définition 4.2
Soient ϕ_1, ϕ_2 deux formules de GCF_{PS}, (c-à-d. formules de choix générales). Soit I une interprétation.

1. $I \mid\sim_k^{PQCL} a$ ssi $k = 1$ et $a \in I$ (pour un atome propositionnel a).

2. $I \mid\sim_k^{PQCL} \neg a$ ssi $k = 1$ et $\neg\, a \in I$ (pour un atome propositionnel a).

3. $I \mid\sim_k^{PQCL} \phi_1 \vec{\times} \phi_2$ ssi

   (a) $I \mid\sim_k^{PQCL} \phi_1$ ou

   (b) $I \mid\sim_n^{PQCL} \phi_2$ et il n'existe pas un entier m tel que $I \mid\sim_m^{PQCL} \phi_1$, et $k = n + opt(\phi_1)$.

4. $I \mid\sim_k^{PQCL} \phi_1 \vee \phi_2$ ssi un des cas suivants est satisfait :

   (a) $(I \mid\sim_1^{PQCL} \phi_1)$ ou $(I \mid\sim_1^{PQCL} \phi_2)$ et k = 1,

   (b) *(Il existe $i > 1$ tel que $I \mid\sim_i^{PQCL} \phi_1$)* et
   $\nexists$ *m tel que* $I \mid\sim_m^{PQCL} \phi_2$, *et* $k = (i\text{-}1) \times opt(\phi_2) + 1$,

   (c) *(Il exist $i > 1$ tel que $I \mid\sim_i^{PQCL} \phi_1$ ou $\nexists$ l, tel que $I \mid\sim_l^{PQCL} \phi_1$)* et
   (il existe $j > 1$ tel que $I \mid\sim_j^{PQCL} \phi_2$), et $k = j$.

5. $I \mid\sim_k^{PQCL} \phi_1 \wedge \phi_2$ ssi $I \mid\sim_i^{PQCL} (\phi_1)$ et $I \mid\sim_j^{PQCL} (\phi_2)$ et $k = (i\text{-}1) \times opt(\phi_2) + j$.

6. $I \mid\sim_k^{PQCL} \neg(\phi_1 \vee \phi_2)$ ssi $I \mid\sim_k^{PQCL} \neg\phi_1 \wedge \neg\phi_2$.

7. $I \mid\sim_k^{PQCL} \neg(\phi_1 \wedge \phi_2)$ ssi $I \mid\sim_k^{PQCL} \neg\phi_1 \vee \neg\phi_2$.

8. $I \mid\sim_k^{PQCL} \neg(\phi_1 \vec{\times} \phi_2)$ ssi $I \mid\sim_k^{PQCL} \neg\phi_1 \vec{\times} \neg\phi_2$.

9. $I \mid\sim_k^{PQCL} \neg(\neg\phi_1)$ ssi $I \mid\sim_k^{PQCL} \phi_1$.

Nous pouvons constater que par rapport à la relation d'inférence $\mathcal{QCL}$ définie dans [Brewka *et al.* 2004], seulement les items (1)-(3) sont similaires. Cela signifie que la relation d'inférence dans le cadre de $\mathcal{PQCL}$ est définie d'une manière identique que celle de $\mathcal{QCL}$ dans le cas où nous traitons uniquement des formules propositionnelles ou des formules de choix de base. Tous les autres items sont différents. En particulier, dans le cadre de la logique $\mathcal{PQCL}$, les définitions de la négation, conjonction et disjonction sont complètement différentes, à l'exception des formules propositionnelles. En effet, comme il sera montré dans les sections qui suivent, la conjonction et la disjonction définies dans $\mathcal{PQCL}$ se comportent comme la conjonction et la disjonction de la logique propositionnelle lorsqu'elles sont appliquées aux formules propositionnelles. Par contre, elles ne sont pas classiques lorsqu'elles sont appliquées aux formules autres

que propositionnelles.
Les items (2), (6), (7), (8) et (9) définissent la relation d'inférence de la négation des formules. Ils indiquent simplement que la négation est décomposable et satisfait également la loi de De Morgan. Les items (4) et (5) définissent respectivement l'inférence de la disjonction et la conjonction qui permettent de représenter des préférences prioritaires. Ainsi, par exemple, les préférences représentées de la forme $(\phi_1 \wedge \phi_2)$ reflètent un certain ordre lexicographique entre les différents degrés de satisfaction de ϕ_1 et ϕ_2. Plus précisément, étant données deux interprétations I, I' alors I est strictement préférée à I' dans l'un des deux cas suivants :

1. $I \mid\sim_i \phi_1$ et $I' \mid\sim_j \phi_1$ avec $i < j$,
2. ($I \mid\sim_i \phi_1$ et $I' \mid\sim_i \phi_1$) et ($I \mid\sim_k \phi_2$ et $I' \mid\sim_l \phi_2$ tel que $k < l$).

Pour résumer, la priorité est premièrement donnée aux préférences représentées par ϕ_1, par la suite les préférences représentées par ϕ_2 sont prises en considération uniquement dans le cas où ϕ_1 est satisfaite par I et I' avec le même degré.

Exemple 4.2
Supposons qu'un agent exprime deux différentes préférences au sujet d'un voyage pendant un week-end. Ces préférences sont données comme suit : *"L'agent préfère Air-France à KLM"* qui pour lui est important que *"Il préfère le siège couloir au siège fenêtre"*. Ces deux préférences sont modélisées et représentées par la formule de choix générale suivante :

$$\phi = (\textit{Air-France} \vec{\times} \textit{KLM}) \wedge (\textit{Couloir} \vec{\times} \textit{Fenêtre})$$

Pour illustrer la relation d'inférence $\mathcal{PQCL}$ pour les préférences de cet agent, nous considérons les interprétations $I_1 = \{KLM, \textit{Fenêtre}\}$ et $I_2 = \{\textit{Air-France}, \textit{couloir}\}$:
La formule ϕ est de la forme $(\phi_1 \wedge \phi_2)$ telles que : $\phi_1 = \textit{Air-France} \vec{\times} KLM$, et $\phi_2 = \textit{Couloir} \vec{\times} \textit{Fenêtre}$.

- $I_1 = \{KLM, \textit{Fenêtre}\}$, nous avons donc $I_1 \mid\sim_{i=2}^{PQCL} \phi_1$ et $I_1 \mid\sim_{j=2}^{PQCL} \phi_2$ (par l'utilisation de l'item (3) de la définition 4.2).
  Nous appliquons l'item (5) de la définition 4.2, nous obtenons $I_1 \mid\sim_k^{PQCL} \phi$ et k = (i-1)×opt(ϕ_2) + j = 4 (correspond à la dernière option de ϕ). Nous concluons que I_1 satisfait ϕ avec le degré 4.
- $I_2 = \{\textit{Air-France}, \textit{couloir}\}$, donc $I_2 \mid\sim_{i=1}^{PQCL} \phi_1$ et $I_2 \mid\sim_{j=1}^{PQCL} \phi_2$ (par l'utilisation de l'item (3) de la définition 4.2).
  Nous appliquons l'item (5) de la définition 4.2, nous obtenons $I_2 \mid\sim_k^{PQCL} \phi$ et k = (i-1)×opt(ϕ_2) + j = 1. Donc, la solution $I_2 = \{\textit{Air-France}, \textit{couloir}\}$ est considérée comme le meilleur choix pour l'agent.

La conjonction et la disjonction dans le cadre de la logique $\mathcal{PQCL}$ sont asymétriques, si ϕ_1 et ϕ_2 sont deux formules dans GCF_{PS}, nous avons :

$$\text{- } \phi_1 \wedge \phi_2 \not\equiv \phi_2 \wedge \phi_1 \text{ et } \phi_1 \vee \phi_2 \not\equiv \phi_2 \vee \phi_1.$$

Ce résultat s'explique par le fait que si un agent exprime plusieurs préférences de la forme $((a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n) \wedge (b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_n))$ (resp. $(a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n) \vee (b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_n)$) (c-à-d. $\phi_1 \wedge \phi_2$ (resp. $\phi_1 \vee \phi_2$)), alors l'ordre n'est pas seulement important au niveau de la disjonction ordonnée (c-à-d. au niveau des options a_i pour ϕ_1 et au niveau des options b_i pour ϕ_2), mais il est

également important au niveau de la conjonction (resp. disjonction). Plus précisément, $\phi_1 \wedge \phi_2$ (resp. $\phi_1 \vee \phi_2$) signifie que ϕ_1 est plus préférée que ϕ_2.

Exemple 4.3
Considérons l'exemple 4.2, où un agent exprime deux différentes préférences sur les alternatives (*Air-France, KLM, Couloir, Fenêtre*) au sujet d'un voyage pendant un week-end. Supposons qu'un autre agent exprime ses préférences sur les mêmes alternatives. Les préférences de chaque agent sont modélisées respectivement par les formules de choix générales ϕ et ψ, telles que ϕ représente les préférences de l'agent 1 et ψ représente les préférences de l'agent 2 :

$\phi = (\textit{Air-France } \vec{\times} \textit{ KLM}) \wedge (\textit{Couloir } \vec{\times} \textit{ Fenêtre})$ et,
$\psi = (\textit{Couloir } \vec{\times} \textit{ Fenêtre}) \wedge (\textit{Air-France } \vec{\times} \textit{ KLM})$.

Par cet exemple, nous illustrons que les préférences de ces deux agents sont différentes dans le sens où ils n'accordent pas la même importance aux alternatives concernées.
Supposons que $I = \{\textit{Air-France, Fenêtre}\}$, les degrés de satisfaction de ϕ et ψ par I sont définis comme suit :
La formule ϕ est de la forme $(\phi_1 \wedge \phi_2)$ tel que $\phi_1 = \textit{Air-France } \vec{\times} \textit{ KLM}$ et $\phi_2 = \textit{Couloir } \vec{\times}$ *Fenêtre*. Par l'application de l'item (3) de la définition 4.2, nous avons $I \mid\!\sim_1^{PQCL} \phi_1$ et $I \mid\!\sim_2^{PQCL} \phi_2$. Nous appliquons l'item (5) de la définition 4.2, nous obtenons $I \mid\!\sim_2^{PQCL} \phi$.
D'un autre coté, la formule ψ est de la forme $(\phi_2 \wedge \phi_1)$ tel que $\phi_2 = \textit{Couloir } \vec{\times} \textit{ Fenêtre}$ et $\phi_1 = \textit{Air-France } \vec{\times} \textit{ KLM}$. Nous avons $I \mid\!\sim_1^{PQCL} \phi_1$ et $I \mid\!\sim_2^{PQCL} \phi_2$, donc $I \mid\!\sim_3^{PQCL} \psi$.

Chacune des deux formules ϕ et ψ, a une optionalité égale à 4. Cependant, leurs degrés de satisfaction sont différents étant donnée l'interprétation I. Et donc, cela confirme que $\phi_1 \wedge \phi_2 \neq \phi_2 \wedge \phi_1$. Ce résultat est valable également pour $\phi_1 \vee \phi_2$ qui n'est pas équivalent à $\phi_2 \vee \phi_1$.

4.2.3 Modèles préférés et inférence à partir d'un ensemble de connaissances et préférences

Soit K un ensemble de formules propositionnelles pour représenter les connaissances de l'agent ou les contraintes d'intégrités, et T soit un ensemble de formules de choix de base ou de choix générales pour représenter les préférences. La définition des modèles préférés dans le cadre de la logique $\mathcal{PQCL}$ est similaire à celle donnée dans [Brewka *et al.* 2004] que nous avons également rappelée dans la définition 4.3 dans le sens où nous utilisons exactement les mêmes procédures en définissant en premier l'optionalité (Définition 2.4 pour $\mathcal{QCL}$ et Définition 4.1 pour $\mathcal{PQCL}$), le degré de satisfaction (Définitions 2.3 et 2.5 pour $\mathcal{QCL}$ et Définitions 2.3 et 4.2 pour $\mathcal{PQCL}$) et les modèles préférés (Définition 2.8 pour $\mathcal{QCL}$ et Définition 4.3 $\mathcal{PQCL}$). L'unique différence (qui est très importante) concerne le degré de satisfaction associé à chaque formule de choix générale dans T. Ainsi, dans le cadre de la logique $\mathcal{QCL}$, la négation d'une formule de choix de base ou de choix générale est équivalente à une formule propositionnelle, où le degré de satisfaction de la formule obtenue (si elle est satisfaite) est toujours égal à 1. Par contre, dans le cadre de la logique $\mathcal{PQCL}$, la notion de préférence est préservée dans la négation d'une formule et son degré de satisfaction est défini comme n'importe quelle formule de choix de base ou de choix générale.

Définition 4.3
Soit $I^k(T)$ un sous ensemble de formules de T satisfaites par un modèle I avec un degré k. Un

modèle I_1 est $K \cup T$ préféré à un modèle I_2 s'il existe un entier k tel que $\mid I_1^k(T)\mid > \mid I_2^k(T)\mid$ et pour tout $j < k : \mid I_1^j(T)\mid = \mid I_2^j(T)\mid$.
I est un modèle préféré de $K \cup T$ ssi I est un $K \cup T$ maximum-préféré.
On dit qu'un modèle I_1 est maximum-préféré s'il n'existe pas un modèle I_2 qui est strictement $K \cup T$-préféré à I_1.

L'exemple suivant illustre le fait que la logique $\mathcal{PQCL}$ surmonte une des limites de $\mathcal{QCL}$ qui sont introduites dans la section 3.2.

Exemple 4.4
Considérons l'exemple 3.1 où la base de connaissances K contient :

$$\phi_1 = \neg \mathit{EasyJet} \vee \neg \mathit{Air\text{-}France}$$

et la base de préférences T contient les préférences suivantes :

$$\begin{cases} \phi_2 = \neg(\mathit{Air\text{-}France} \vec{\times} \mathit{EasyJet}) \vee \mathit{Formule\text{-}Hotel} \\ \phi_3 = \neg(\mathit{EasyJet} \vec{\times} \mathit{Air\text{-}France}) \vee \neg \mathit{Formule\text{-}Hotel} \\ \phi_4 = \mathit{Air\text{-}France} \vec{\times} \mathit{EasyJet} \end{cases}$$

Pour calculer les modèles préférés de la base $K \cup T$, nous procédons en premier à calculer le degré de satisfaction de chacune des formules ϕ_1, ϕ_2, ϕ_3 et ϕ_4 pour toute interprétation.

Par exemple, soit $I = \{\mathit{Air\text{-}France}, \mathit{Formule\text{-}Hotel}\}$, et soit la préférence ϕ_2 (c-à-d. $\neg$ (*Air-France* $\vec{\times}$ *EasyJet*) $\vee$ *Formule-Hotel*). Par l'application de l'item (8) de la définition 4.2, nons obtenons : $I \mid\sim_2^{PQCL} \neg$ (*Air-France* $\vec{\times}$ *EasyJet*), et $I \mid\sim_1^{PQCL}$ *Formule-Hotel*,
Utilisons l'item 4-(a) de la définition 4.2, nous obtenons :
$I \mid\sim_1^{PQCL}$ ($\neg$ *Air-France* $\vec{\times}$ *EasyJet*) $\vee$ *Formule-Hotel*.
D'une manière similaire, nous avons également $I \mid\sim_1^{PQCL} \neg$(*EasyJet* $\vec{\times}$ *Air-France*)$\vee\neg$ *Formule-Hotel*, du fait que $I \mid\sim_1^{PQCL} \neg$(*EasyJet* $\vec{\times}$ *Air-France*).

Le tableau suivant donne le degré de satisfaction de chaque formule ϕ_1, ϕ_2, ϕ_3 et ϕ_4 de K et T. Pour toute interprétation, le nombre (1, 2) indique le degré de satisfaction de la formule si elle satisfaite, ou (-) si elle est falsifiée.

Air-France	*EasyJet*	*Formule-Hotel*	ϕ_1	ϕ_2	ϕ_3	ϕ_4
F	F	F	1	1	1	-
F	F	T	1	1	1	-
F	T	F	1	1	1	2
F	T	T	1	1	2	2
T	F	F	1	2	1	1
T	**F**	**T**	**1**	**1**	**1**	**1**
T	T	F	-	-	1	1
T	T	T	-	1	-	1

TAB. 4.1: Les modèles de la base $K \cup T$ dans le cadre de la relation d'inférence $\mathcal{PQCL}$

Dans le cadre de la logique $\mathcal{QCL}$, $K \cup T$ est incohérente (comme nous l'avons montré dans l'exemple 3.1). Avec la logique $\mathcal{PQCL}$, $K \cup T$ admet un seul modèle préféré (ligne en gras), $I =$

{*Air-France*, *Formule-Hotel*}, d'où le résultat $K \cup T \mid\!\sim^{PQCL}$ *Formule-Hotel*, qui correspond au résultat attendu.

4.2.4 La fonction de normalisation de la logique $\mathcal{PQCL}$

Dans cette section, nous montrons que le cadre de la logique $\mathcal{PQCL}$ permet également de transformer d'une manière équivalente tout ensemble de préférences représentées par des formules de choix générales en un ensemble de préférences en forme normales représentées par des formules de choix de base. La *fonction de normalisation* de $\mathcal{PQCL}$, notée par $\mathcal{N}_{\mathcal{PQCL}}$ est définie comme suit :

Définition 4.4
Une fonction de normalisation notée par $\mathcal{N}_{\mathcal{PQCL}} : GCF_{PS} \rightarrow BCF_{PS}$, *est une fonction qui affecte une formule de choix de base pour toute formule de choix générale, tel que :*

1. *La forme normale de toute formule de choix de base ou formule propositionnelle correspond à la formule elle même :*
   (a) $\forall \phi \in BCF_{PS}, \mathcal{N}_{\mathcal{PQCL}}(\phi) = \phi$.

2. *La forme normale est décomposable respectivement par rapport à la négation, conjonction, disjonction et disjonction ordonnée des formules de choix générales :*
   (a) $\forall \phi \in GCF_{PS}$ *et* $\phi \notin BCF_{PS}$,
   $\mathcal{N}_{\mathcal{PQCL}}(\neg\phi) \equiv \mathcal{N}_{\mathcal{PQCL}}(\neg\mathcal{N}_{\mathcal{PQCL}}(\phi))$.

   (b) $\forall \phi, \psi \in GCF_{PS}$ *et* ($\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS}$),
   $\mathcal{N}_{\mathcal{PQCL}}(\phi \wedge \psi) \equiv \mathcal{N}_{\mathcal{PQCL}}(\mathcal{N}_{\mathcal{PQCL}}(\phi) \wedge \mathcal{N}_{\mathcal{PQCL}}(\psi))$.

   (c) $\forall \phi, \psi \in GCF_{PS}$ *et* ($\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS}$),
   $\mathcal{N}_{\mathcal{PQCL}}(\phi \vee \psi) \equiv \mathcal{N}_{\mathcal{PQCL}}(\mathcal{N}_{\mathcal{PQCL}}(\phi) \vee \mathcal{N}_{\mathcal{PQCL}}(\psi))$.

   (d) $\forall \phi, \psi \in GCF_{PS}$ *et* ($\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS}$),
   $\mathcal{N}_{\mathcal{PQCL}}(\phi \vec{\times} \psi) \equiv \mathcal{N}_{\mathcal{PQCL}}(\mathcal{N}_{\mathcal{PQCL}}(\phi) \vec{\times} \mathcal{N}_{\mathcal{PQCL}}(\psi))$.

3. *La normalisation de la négation, conjonction et disjonction des formules de choix de base est donnée comme suit :*

   (a) *Soient* $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, *et* $\psi = b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m$ *telles que les* a_i *et les* b_i *sont des formules propositionnelles,*
   $\mathcal{N}_{\mathcal{PQCL}}(\phi \wedge \psi) \equiv c_{11} \vec{\times} \ldots \vec{\times} c_{1m} \vec{\times} c_{21} \vec{\times} \ldots \vec{\times} c_{2m} \vec{\times} \ldots \vec{\times} c_{n1} \vec{\times} \ldots \vec{\times} c_{nm}$, *avec* $c_{ij} = a_i \wedge b_j$.

   (b) *Soient* $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, *et* $\psi = b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m$ *telles que les* a_i *et les* b_i *sont des formules propositionnelles,*
   $\mathcal{N}_{\mathcal{PQCL}}(\phi \vee \psi) \equiv d_{11} \vec{\times} \ldots \vec{\times} d_{1m} \vec{\times} d_{21} \vec{\times} \ldots \vec{\times} d_{2m} \vec{\times} \ldots \vec{\times} d_{n1} \vec{\times} \ldots \vec{\times} d_{nm}$, *avec* $d_{ij} = a_i \vee b_j$.

(c) Soit $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ *telles que les* a_i *sont des formules propositionnelles,* $\mathcal{N}_{\mathcal{PQCL}}(\neg\phi) \equiv \neg a_1 \vec{\times} \neg a_2 \vec{\times} \ldots \vec{\times} \neg a_n$.

La propriété (1) indique que la forme normale d'une formule de choix de base ϕ, correspond à la formule ϕ. La propriété 2 (a, b, c, d) exprime que la forme normale est décomposable par rapport à la négation, conjonction, disjonction et la disjonction ordonnée. La propriété 3 (a, b, c) donne la définition de la conjonction, disjonction et négation appliquée aux formules de choix de base. La propriété 3-(a) définit la conjonction prioritaire. En effet, supposons que $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ représente les préférences d'un utilisateur A et $b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m$ représente les préférences de B. L'application de la conjonction prioritaire permet de sélectionner des solutions qui privilégient A. Par exemple, $a_1 b_m$ (qui représente le meilleur choix pour A et le pire choix pour B) est préféré à $a_2 b_1$ (qui représente le meilleur choix pour B et le second choix pour A).

Dans l'exemple suivant, nous illustrons l'utilisation d'une fonction de normalisation sur une formule de choix générale dans GCF_{PS}.

Exemple 4.5
Soit $F = \neg(\text{a } \vec{\times} \text{ b}) \vee ((\text{c } \vec{\times} \neg \text{ d}) \wedge \text{e})$ une formule de choix générale. Pour transformer F en forme normale, nous utilisons la définition 4.4. La formule F est une disjonction de deux formules de choix générales (F_1 et F_2) tel que $F_1 = \neg(\text{a } \vec{\times} \text{ b})$ et $F_2 = (\text{c } \vec{\times} \neg \text{ d}) \wedge \text{e}$. Nous appliquons en premier l'item 2-(c) de la définition 4.4 tel que :
$\mathcal{N}_{\mathcal{PQCL}}(\neg(\text{a } \vec{\times} \text{ b}) \vee ((\text{c } \vec{\times} \neg \text{ d}) \wedge \text{e})) \equiv \mathcal{N}_{\mathcal{PQCL}}(\mathcal{N}_{\mathcal{PQCL}}(\neg(\text{a } \vec{\times} \text{ b})) \vee \mathcal{N}_{\mathcal{PQCL}}((\text{c } \vec{\times} \neg \text{ d}) \wedge \text{e}))$.
Par la suite, nous appliquons l'item 3-(c) de la définition 4.4 pour normaliser F_1. Ainsi, le résultat donne $\neg a \vec{\times} \neg b$ qui est une formule de choix de base, c-à-d. $\mathcal{N}_{\mathcal{PQCL}}(\neg a \vec{\times} \neg b) = \neg a \vec{\times} \neg b$. D'un autre coté, F_2 est une conjonction de deux formules, la première étant une formule de choix de base et la seconde est propositionnelle. Donc, nous appliquons l'item 3-(a) de la définition 4.4, la normalisation de F_2 donne $(\text{c} \wedge \text{e}) \ \vec{\times} (\neg \text{ d} \wedge \text{e})$. Finalement, nous pouvons appliquer l'item 3-(b) de la définition 4.4 pour normaliser la formule obtenue (disjonction de deux formules de choix de base). Ces différentes étapes sont détaillées comme suit :

$\mathcal{N}_{\mathcal{PQCL}}(\neg(\text{a } \vec{\times} \text{ b}) \vee ((\text{c } \vec{\times} \neg \text{ d}) \wedge \text{e})) \equiv \mathcal{N}_{\mathcal{PQCL}}(\mathcal{N}_{\mathcal{PQCL}}(\neg(\text{a } \vec{\times} \text{ b})) \vee \mathcal{N}_{\mathcal{PQCL}}((\text{c} \vec{\times} \neg \text{ d}) \wedge \text{e}))$
$\equiv \mathcal{N}_{\mathcal{PQCL}}((\neg \text{ a } \vec{\times} \neg \text{ b}) \vee ((\text{c} \wedge \text{e}) \ \vec{\times} \ (\neg \text{ d} \wedge \text{e})))$
$\equiv \mathcal{N}_{\mathcal{PQCL}}((\neg \text{ a} \vee (\text{c} \wedge \text{e})) \ \vec{\times} \ (\neg \text{ a} \vee (\neg \text{ d} \wedge \text{e})) \ \vec{\times} \ (\neg \text{ b} \vee (\text{c} \wedge \text{e})) \ \vec{\times} \ (\neg \text{ b} \vee (\neg \text{ d} \wedge \text{e}))$
$\equiv (\neg \text{ a} \vee (\text{c} \wedge \text{e})) \ \vec{\times} \ (\neg \text{ a} \vee (\neg \text{ d} \wedge \text{e})) \ \vec{\times} \ (\neg \text{ b} \vee (\text{c} \wedge \text{e})) \ \vec{\times} \ (\neg \text{ b} \vee (\neg \text{ d} \wedge \text{e}))$.

La proposition suivante montre que la relation d'inférence $\mid\!\sim^{PQCL}$ peut être également donnée par l'utilisation de la fonction de normalisation :

Proposition 4.1
Soient T *un ensemble de formules* $\mathcal{PQCL}$ *(GCF ou BCF), et* K *un ensemble de formules propositionnelles. Soit* T' *un ensemble de formules de choix de base obtenu de* T *par le remplacement de chaque formule* ϕ *dans* T *par* $\mathcal{N}_{\mathcal{PQCL}}(\phi)$*, alors* $\forall \psi \in PROP_{PS}$, $K \cup T \mid\!\sim^{PQCL} \psi$ *ssi* $K \cup T' \mid\!\sim^{PQCL} \psi$.
Ainsi, nous proposons deux possibilités pour définir l'inférence de chaque formule dans GCF_{PS} :

1. Utilisation de la relation d'inférence des formules de choix générales, comme indiqué dans la définition 4.2,
2. Normaliser ou générer l'ensemble des formules de choix de base à partir de tout ensemble de préférences (Définition 4.4), nous appliquons par la suite, la relation de satisfaction des formules BCF comme c'est indiqué dans la définition 2.3.

Preuve de la proposition 4.1

Pour montrer cette proposition, il suffit de considérer deux formules de choix de base $\phi_1 = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, $\phi_2 = b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m$, et montrer que $\forall I \in \Omega$, $\forall k$ (Ω représente l'ensemble de toutes les interprétations classiques), nous avons :

- $I \mid\sim_k^{PQCL} (\phi_1 \vec{\times} \phi_2)$ ssi $I \models_k \mathcal{N}_{\mathcal{PQCL}}(\phi_1 \vec{\times} \phi_2)$,
- $I \mid\sim_k^{PQCL} (\phi_1 \wedge \phi_2)$ ssi $I \models_k \mathcal{N}_{\mathcal{PQCL}}(\phi_1 \wedge \phi_2)$,
- $I \mid\sim_k^{PQCL} (\phi_1 \vee \phi_2)$ ssi $I \models_k \mathcal{N}_{\mathcal{PQCL}}(\phi_1 \vee \phi_2)$.

Nous commençons en premier par la preuve de $I \mid\sim_k^{PQCL} (\phi_1 \vec{\times} \phi_2)$ ssi $I \models_k \mathcal{N}_{\mathcal{PQCL}}(\phi_1 \vec{\times} \phi_2)$.

1. Rappelons que : $\mathcal{N}_{\mathcal{PQCL}}(\phi_1 \vec{\times} \phi_2) \equiv\ a_1 \vec{\times} \ldots \vec{\times} a_n \vec{\times} b_1 \vec{\times} \ldots \vec{\times} b_m$. Nous considérons trois cas :
   - Supposons qu'il existe un $i > 0$ tel que $I \models_i\ a_1 \vec{\times} \ldots \vec{\times} a_n$. Cela signifie que I falsifie chacune des propositions $\{ a_1, \ldots, a_{i-1}\}$, mais I satisfait a_i. Nous avons (i-1) options qui ne sont pas satisfaites avant que l'option (a_i) soit satisfaite par I. Donc, cela signifie que $I \models_i\ a_1 \vec{\times} \ldots \vec{\times} a_n \vec{\times} b_1 \vec{\times} \ldots \vec{\times} b_m$.
     Donc, $I \models_i \phi_1 \vec{\times} \phi_2$. Par l'utilisation de l'item 3-(a) de la définition 4.2, nous pouvons vérifier que nous avons également $I \mid\sim_k^{PQCL} \phi_1 \vec{\times} \phi_2$ et k = i.

   - Supposons qu'il n'existe pas un i tel que $I \not\models\ a_1 \vec{\times} \ldots \vec{\times} a_n$, et il existe un j tel que $I \models_j\ b_1 \vec{\times} \ldots \vec{\times} b_m$. Cela signifie que I falsifie chacune des propositions $\{ a_1, \ldots, a_{i-1}, ..., a_n, b_1, \ldots, b_{j-1}\}$, mais I satisfait b_j. Nous avons $n + (j-1)$ options qui ne sont pas satisfaites avant que l'option (b_j) soit satisfaite. Donc, cela signifie que $I \models_k\ a_1 \vec{\times} \ldots \vec{\times} a_n \vec{\times} b_1 \vec{\times} \ldots \vec{\times} b_m$, c'est-à-dire que $I \models_k \phi_1 \vec{\times} \phi_2$ et $k = n + j$. Nous utilisons l'item 3-(b) de la définition 4.2, donc nous avons aussi $I \mid\sim_k^{PQCL} \phi_1 \vec{\times} \phi_2$, et k = opt(ϕ_1) + j = n + j.

   - Si il n'existe ni un i, ni un j, tel que $I \not\models\ a_1 \vec{\times} \ldots \vec{\times} a_n$ et $I \not\models\ b_1 \vec{\times} \ldots \vec{\times} b_m$. Cela signifie que I falsifie toutes les options a_i, b_j, ainsi $I \not\models (\phi_1 \vec{\times} \phi_2)$.
     Et nous avons bien $I \not\mid\sim^{PQCL} \phi_1 \vec{\times} \phi_2$.

2. Dans ce qui suit, nous donnons la preuve de $I \mid\sim_k^{PQCL} (\phi_1 \wedge \phi_2)$ ssi $I \models_k \mathcal{N}_{\mathcal{PQCL}}(\phi_1 \wedge \phi_2)$.
   Rappelons que :
   $\mathcal{N}_{\mathcal{PQCL}}(\phi_1 \wedge \phi_2) \equiv\ c_{11} \vec{\times} \ldots \vec{\times} c_{1m} \vec{\times} c_{21} \vec{\times} \ldots \vec{\times} c_{2m} \vec{\times} \ldots \vec{\times} c_{n1} \vec{\times} \ldots \vec{\times} c_{nm}$, avec $c_{ij} = a_i \wedge b_j$.
   Nous considérons deux cas concernant la satisfaction des $a_{i=1,\ldots,n}$ et $b_{j=1,\ldots m}$ par l'interprétation I.

   - Supposons qu'il existe un $i > 0$ et un $j > 0$ tel que $I \models_i\ a_1 \vec{\times} \ldots \vec{\times} a_n$ et $I \models_j\ b_1 \vec{\times} \ldots \vec{\times} b_m$. Cela signifie que $I \models \neg a_1 \wedge ... \wedge \neg a_{i-1} \wedge a_i$ et $I \models \neg b_1 \wedge ... \wedge \neg b_{j-1} \wedge b_j$.
     Cela signifie également que I falsifie $\{ c_{11}, \ldots, c_{1m}, c_{21}, \ldots, c_{2m}, \ldots, c_{i1}, \ldots, c_{i(j-1)}\}$, mais I satisfait $c_{ij} (= a_i \wedge b_j)$. Nous avons donc, ($i$-1)$\times m + j - 1$ options qui ne sont pas satisfaites avant que ($a_i \wedge b_j$) soit satisfaite. Ainsi, cela signifie que $I \models_k\ c_{11} \vec{\times} \ldots \vec{\times} c_{1m} \vec{\times} c_{21} \vec{\times} \ldots \vec{\times} c_{2m} \vec{\times} c_{n1} \vec{\times} \ldots \vec{\times} c_{nm}$, c'est-à-dire que $I \models_k \phi_1 \wedge \phi_2$ et $k = (i-1) \times m + j$.
     L'application de l'item (5) de la définition 4.2, nous pouvons vérifier que nous avons aussi $I \mid\sim_k^{PQCL} \phi_1 \wedge \phi_2$ et $k = (i-1) \times$ opt(ϕ_2) $+ j$.

   - Il n'existe pas un i tel que $I \models_i\ a_1 \vec{\times} \ldots \vec{\times} a_n$ ou il n'existe pas un j tel que $I \models_j$

$b_1 \vec{\times} \ldots \vec{\times} b_m$. Cela signifie que soit I falsifie tous les $a_{i=1,\ldots,n}$, soit I falsifie tous les $b_{j=1,\ldots,m}$. Donc, I falsifie les c_{ij} ($= a_i \wedge b_j$), c'est-à-dire qu'il n'existe pas k tel que $I \models_k (\phi_1 \wedge \phi_2)$.
Donc, $I \not\sim^{PQCL} \phi_1 \wedge \phi_2$

3. Dans ce qui suit, nous donnons la preuve de $I \sim_k^{PQCL} (\phi_1 \vee \phi_2)$ ssi $I \models_k \mathcal{N}_{\mathcal{PQCL}}(\phi_1 \vee \phi_2)$.
Rappelons que : $\mathcal{N}_{\mathcal{PQCL}}(\phi_1 \vee \phi_2) \equiv d_{11} \vec{\times} \ldots \vec{\times} d_{1m} \vec{\times} d_{21} \vec{\times} \ldots \vec{\times} d_{2m} \vec{\times} \ldots \vec{\times} d_{n1} \vec{\times} \ldots \vec{\times} d_{nm}$, avec $d_{ij} = a_i \vee b_j$.
Nous considérons différents cas concernant la satisfaction des $a_{i=1,\ldots,n}$ et $b_{j=1,\ldots,m}$ par l'interprétation I.

   – Supposons qu'il existe un $i > 0$ ou un $j > 0$ tel que $I \models_i a_1 \vec{\times} \ldots \vec{\times} a_n$ et $I \models_j b_1 \vec{\times} \ldots \vec{\times} b_m$. Cela signifie qu'il existe un $k > 0$ tel que $I \models_k d_k$. Nous pouvons distinguer différents cas :

     • Si $i = 1$ ou $j = 1$, alors I satisfait $\{ d_{11}, d_{12}, \ldots, d_{1(j-1)}, d_{1j}, \ldots, d_{1m}, d_{21}, \ldots d_{i1}\}$. Dans ce cas, l'option qui est satisfaite en premier est d_{11}, d'où
     $I \models_1 d_{11} \vec{\times} \ldots \vec{\times} d_{1m} \vec{\times} d_{21} \vec{\times} \ldots \vec{\times} d_{2m} \vec{\times} \ldots \vec{\times} d_{n1} \vec{\times} \ldots \vec{\times} d_{nm}$. Donc $I \models_1 (\phi_1 \vee \phi_2)$. Comme $I \models_1 (\phi_1 \vee \phi_2)$ et nous avons supposé qu'il existe un i = 1 qui satisfait $a_1 \vec{\times} \ldots \vec{\times} a_n$, c'est-à-dire ϕ_1 ou il existe un j = 1 qui satisfait $b_1 \vec{\times} \ldots \vec{\times} b_n$, c'est-à-dire ϕ_2, nous utilisons maintenant l'item 4-(a) de la définition 4.2, nous pouvons vérifier que nous avons aussi $I \sim_k^{PQCL} \phi_1 \vee \phi_2$ et $k = 1$ car nous avons $I \models_1 \phi_1$ ou $I \models_1 \phi_2$.

     • Si $i > 1$ ou $j > 0$, alors I satisfait au moins les options $\{ d_{1j}, \ldots, d_{2j}, \ldots, d_{ij}\}$.
     Dans ce cas, nous avons $(j-1)$ options qui ne sont pas satisfaites avant l'option d_{1j} qui est satisfaite, d'où $I \models_j d_{11} \vec{\times} \ldots \vec{\times} d_{1m} \vec{\times} d_{21} \vec{\times} \ldots \vec{\times} d_{2m} \vec{\times}$
     $\ldots \vec{\times} d_{n1} \vec{\times} \ldots \vec{\times} d_{nm}$. Donc, $I \models_j (\phi_1 \vee \phi_2)$.
     D'un autre coté, comme d_{1j} (c-à-d. $a_1 \vee b_j$) est la première option qui est satisfaite par I, donc soit a_1 est satisfaite ou soit b_j. Or, a_1 n'est pas satisfaite car nous avons supposé que $i > 1$, donc c'est b_j qui est satisfaite et cela signifie que $I \models_j \phi_2$.
     Donc, si j=1, nous utilisons l'item 4-(a) de la définition 4.2, nous avons $I \sim_k^{PQCL} \phi_1 \vee \phi_2$ et $k = 1$. si $j > 1$, nous utilisons l'item 4-(c) de la définition 4.2, nous avons $I \sim_k^{PQCL} \phi_1 \vee \phi_2$ et $k = j$.

   – Il n'existe pas de i tel que $I \models_i a_1 \vec{\times} \ldots \vec{\times} a_n$ et il existe un j tel que $I \models_j b_1 \vec{\times} \ldots \vec{\times} b_m$. Cela signifie que $I \models \neg a_1 \wedge \ldots \wedge \neg a_{i-1} \wedge \neg a_i$ et $\exists\, j > 0$ tel que $I \models \neg b_1 \wedge \ldots \wedge \neg b_{j-1} \wedge b_j$. Cela signifie également que I falsifie $\{ d_{11}, \ldots, d_{1(j-1)}, \ldots, d_{21}, \ldots, d_{2(j-1)} \ldots, d_{(i-1)(j-1)}\}$, mais I satisfait les options $\{d_{1j}, d_{2j}, \ldots, d_{(i-1)j}, d_{ij}\}$. Ainsi, d_{1j} est l'option qui est satisfaite en premier et toutes les options avant d_{1j} ne sont pas satisfaites, c'est-à-dire que nous avons au moins $(j-1)$ options qui ne sont pas satisfaites avant la première option qui est satisfaite $(a_1 \vee b_j)$, cela signifie que :
   $I \models_k d_{11} \vec{\times} \ldots \vec{\times} d_{1m} \vec{\times} d_{21} \vec{\times} \ldots \vec{\times} d_{2m} \vec{\times} \ldots \vec{\times} d_{n1} \vec{\times} \ldots \vec{\times} d_{nm}$, donc $I \models_j (\phi_1 \vee \phi_2)$.
   Comme, il n'existe pas de i qui satisfait ϕ_1, ϕ_2 est donc satisfaite avec le degré j. Si j > 1, nous utilisant l'item 4-(c) de la définition 4.2, nous avons $I \sim_k^{PQCL} (\phi_1 \vee \phi_2)$, et $k = j$. Si j = 1, nous utilisant l'item 4-(a) de la définition 4.2, nous avons $I \sim_k^{PQCL} (\phi_1 \vee \phi_2)$, et $k = 1$.

   – Il existe un i tel que $I \models_i a_1 \vec{\times} \ldots \vec{\times} a_n$ et il n'existe pas un j tel que $I \models_j b_1 \vec{\times} \ldots \vec{\times} b_m$.

Cela signifie qu'il existe un $i > 0$ tel que $I \models \neg a_1 \wedge ... \wedge \neg a_{i-1} \wedge a_i$ et il n'existe pas un j tel que $I \models \neg b_1 \wedge ... \wedge \neg b_{j-1} \wedge \neg b_j$.
Cela signifie que I falsifie les options $\{ d_{11}, \ldots, d_{1(j-1)}, d_{1j}..., d_{21}, \ldots, d_{2(j-1)}, d_{2j},..., d_{(i-1)(j-1)}\}$, mais I satisfait les options $\{d_{i1}, ..., d_{(i)(j-1)}, d_{ij}\}$. Ainsi, d_{i1} est l'option qui est satisfaite en premier et toutes les options avant d_{i1} $(=a_i \vee b_1)$ ne sont pas satisfaites, c'est-à-dire, nous avons au moins $(i-1)\times$m options qui ne sont pas satisfaites, cela signifie que $I \models_k d_{11}\vec{\times}\ldots\vec{\times} d_{1m}\vec{\times} d_{21}\vec{\times}\ldots\vec{\times} d_{2m}\vec{\times}\ldots\vec{\times} d_{n1}\vec{\times}\ldots\vec{\times} d_{nm}$, ainsi $I \models_k (\phi_1 \vee \phi_2)$, et k = (i-1)×m +1.
Donc, utilisons l'item 4-(b) de la définition 4.2, nous avons aussi :
$I \mid\sim_k^{PQCL} (\phi_1 \vee \phi_2)$, et k = (i-1)×opt(ϕ_2) +1.

– Il n'existe pas un i tel que $I \models_i a_1\vec{\times}\ldots\vec{\times} a_n$ et il n'existe pas un j tel que $I \models_j b_1\vec{\times}\ldots\vec{\times} b_m$. Cela signifie que I soit elle falsifiait toutes les options $a_{i=1,...,n}$, soit elle falsifie toutes les options $b_{j=1,...m}$. D'où, I falsifie toutes les options d_{ij} (= $a_i \vee b_j$).
Donc, il n'existe pas k tel que $I \models_k (\phi_1 \vee \phi_2)$, ainsi $I \not\mid\sim^{PQCL} (\phi_1 \vee \phi_2)$. □

Dans ce qui suit, nous illustrons la proposition 4.1 par la formule de choix générale donnée dans l'exemple 4.5. Nous utilisons pour cela les deux différentes méthodes que nous avons proposées pour définir le degré de satisfaction de cette formule. Soit l'interprétation I = {a,e}.

1. Normalisation de l'ensemble de préférences en un ensemble de formules de choix de base :
Comme nous l'avons montré dans l'exemple 4.5, la formule $F = \neg(a\vec{\times}b) \vee ((c\vec{\times}\neg d) \wedge e)$ est de choix générale et $(\neg a \vee (c \wedge e))\vec{\times}(\neg a \vee (\neg d \wedge e))\vec{\times}(\neg b \vee (c \wedge e))\vec{\times}(\neg b \vee (\neg d \wedge e))$ est la formule de choix de base obtenue par l'application de la fonction de normalisation $\mathcal{N}_{\mathcal{PQCL}}$.
Par l'application de la définition 2.3, nous obtenons $I \not\models \neg a \vee (c \wedge e)$, $I \models \neg a \vee (\neg d \wedge e)$, $I \models \neg b \vee (c \wedge e)$ et $I \models \neg b \vee (\neg d \wedge e)$, donc $I \mid\sim_2^{PQCL} F$.

2. Utiliser uniquement la relation d'inférence $\mathcal{PQCL}$ (Définition 4.2) :
La formule F est de la forme $(F_1 \vee F_2)$ tel que $F_1 = \neg$ (a $\vec{\times}$ b) et F_2 = (c $\vec{\times}\neg$ d) ∧ e.
Par l'utilisation de l'item (3) de la définition 4.2, nous avons $I \mid\sim_{i=1}^{PQCL} a\vec{\times}b$, et lorsque nous appliquons l'item (8) de la définition 4.2, nous obtenons :
$I \mid\sim_{i=2}^{PQCL} \neg(a\vec{\times}b)$.
La formule F_2 est de la forme $F' \wedge F''$ tel que $F' = c\vec{\times}\neg d$ et $F'' = e$. Donc, nous avons $I \mid\sim_{j'=2}^{PQCL} c\vec{\times}\neg d$ et $I \mid\sim_{j''=1}^{PQCL} e$, alors $I \mid\sim_j^{PQCL} F_2$ et j = (j'-1)×$opt(F'')$ + j'' = (2-1)× 1 + 1 = 2 (utilisant l'item (5) de la définition 4.2).
Finalement, nous appliquons l'item 4-(c) de la définition 4.2, nous obtenons ainsi :
$I \mid\sim_k^{PQCL} F$ et $k = j = 2$.
Nous concluons que le résultat obtenu est le même dans les deux cas.

Dans la section suivante, nous présentons la troisième alternative qui concerne le traitement des préférences positives.

4.3 $\mathcal{QCL}+$: Logique du Choix Qualitatif pour le traitement des préférences positives

Dans cette section, nous proposons une troisième alternative permettant de surmonter les limites de la logique $\mathcal{QCL}$. Il s'agit d'une nouvelle logique appelée *Logique du Choix Qualitatif positif* et notée ($\mathcal{QCL}+$) [Benferhat & Sedki 2008b] qui est appropriée pour le traitement des préférences positives. La logique $\mathcal{QCL}+$ partage de nombreuses caractéristiques que la logique $\mathcal{PQCL}$. Elle est basée sur le même langage $\mathcal{QCL}$ dans le sens où les préférences sont représentées par des formules de choix de base ou de choix générales et les connaissances sont représentées par des formules propositionnelles. La sémantique de toute formule dans le cadre de $\mathcal{QCL}+$ est également basée sur son degré de satisfaction dans un modèle particulier I. La négation dans le cadre de la logique $\mathcal{QCL}+$ est similaire à celle de $\mathcal{PQCL}$, par conséquent, en $\mathcal{QCL}+$ la double négation d'une formule est équivalente à la formule originale. Toutefois, la définition de la conjonction (resp. disjonction) n'est pas prioritaire. En fait, dans le cadre de la logique $\mathcal{QCL}+$, la conjonction et la disjonction définies sont plus adéquates pour la modélisation des préférences positives. Les préférences sont dites positives lorsqu'elles n'excluent pas de solutions dans la mesure où uniquement ce qui souhaitable est exprimé. Pour pouvoir modéliser correctement de telles préférences positives, nous avons légèrement modifier la notion des modèles préférés.
$\mathcal{QCL}+$ est caractérisée par une fonction de normalisation, notée $\mathcal{N}_{\mathcal{QCL}+}$, qui partage certaines caractéristiques de la fonction de normalisation de la logique $\mathcal{PQCL}$, telles que la décomposabilité par rapport à la négation, conjonction, disjonction et la disjonction ordonnée, par ailleurs, concernant la conjonction et la disjonction, elle fournit de nouvelles définitions comme nous allons le définir avec plus de précision dans ce qui suit.

4.3.1 Normalisation des préférences dans le cadre $\mathcal{QCL}+$

La forme normale des préférences positives dans le cadre de la logique $\mathcal{QCL}+$ est définie comme suit :

Définition 4.5
Dans le cadre de la logique $\mathcal{QCL}+$, nous appelons fonction de normalisation, *notée par $\mathcal{N}_{\mathcal{QCL}+}$, une fonction de $GCF_{PS} \rightarrow BCF_{PS}$, tel que :*

1. *La forme normale de toute formule de choix de base ou formule propositionnelle correspond à la même formule :*
   (a) $\forall \phi \in BCF_{PS}, \mathcal{N}_{\mathcal{QCL}+}(\phi) = \phi$.

2. *La forme normale est décomposable par rapport à la négation, conjonction, disjonction et la disjonction ordonnée des formules de choix générales :*
   (a) $\forall \phi \in GCF_{PS}$ *et* $\phi \notin BCF_{PS}$,
   $\mathcal{N}_{\mathcal{QCL}+}(\neg\phi) \equiv \mathcal{N}_{\mathcal{QCL}+}(\neg\mathcal{N}_{\mathcal{QCL}+}(\phi))$.

   (b) $\forall \phi, \psi \in GCF_{PS}$ *et* ($\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS}$),
   $\mathcal{N}_{\mathcal{QCL}+}(\phi \wedge \psi) \equiv \mathcal{N}_{\mathcal{QCL}+}(\mathcal{N}_{\mathcal{QCL}+}(\phi) \wedge \mathcal{N}_{\mathcal{QCL}+}(\psi))$.

   (c) $\forall \phi, \psi \in GCF_{PS}$ *et* ($\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS}$),
   $\mathcal{N}_{\mathcal{QCL}+}(\phi \vee \psi) \equiv \mathcal{N}_{\mathcal{QCL}+}(\mathcal{N}_{\mathcal{QCL}+}(\phi) \vee \mathcal{N}_{\mathcal{QCL}+}(\psi))$.

(d) $\forall \phi, \psi \in GCF_{PS}$ *et* ($\phi \notin BCF_{PS}$ *ou* $\psi \notin BCF_{PS}$),
$\mathcal{N}_{\mathcal{QCL}+}(\phi \vec{\times} \psi) \equiv \mathcal{N}_{\mathcal{QCL}+}(\mathcal{N}_{\mathcal{QCL}+}(\phi) \vec{\times} \mathcal{N}_{\mathcal{QCL}+}(\psi))$.

3. *La forme normale de la négation, conjonction et disjonction des formules de choix de base est :*
Soient $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, *et* $\psi = b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_m$ *telles que les* a_i *et les* b_i *sont des formules propositionnelles.*

(a) $\mathcal{N}_{\mathcal{QCL}+}(\phi \wedge \psi) \equiv c_1 \vec{\times} c_2 \vec{\times} \ldots \vec{\times} c_k$ *où* $k = max(m, n)$, *et*

$$c_i = \begin{cases} (a_i \wedge b_i) & si \quad i \leq min(m,n) \\ a_i & si \quad m \leq i \leq n \\ b_i & si \quad n \leq i \leq m \end{cases}$$

(b) $\mathcal{N}_{\mathcal{QCL}+}(\phi \vee \psi) \equiv c_1 \vec{\times} c_2 \vec{\times} \ldots \vec{\times} c_k$ *où* $k = max(m, n)$, *et*

$$c_i = \begin{cases} (a_i \vee b_i) & si \quad i \leq min(m,n) \\ a_i & si \quad m \leq i \leq n \\ b_i & si \quad n \leq i \leq m \end{cases}$$

(c) $\mathcal{N}_{\mathcal{QCL}+} \neg(\ a_1 \vec{\times}\ a_2 \vec{\times} \ldots \vec{\times}\ a_n) \equiv \neg\ a_1 \vec{\times} \neg\ a_2 \vec{\times} \ldots \vec{\times} \neg\ a_n$.

Intuitivement, étant donné un agent qui exprime des préférences que nous représentons par la formule $(\phi_1 \wedge \phi_2)$ dans le cadre de la logique $\mathcal{QCL}+$, la solution la plus préférée concerne celle qui satisfait la première option de ϕ_1 et également la première option de ϕ_2. La seconde solution est celle qui satisfait la seconde option de ϕ_1 et la seconde option de ϕ_2, etc.

La conjonction et la disjonction dans le cadre de la logique $\mathcal{QCL}+$ sont symétriques, si ϕ_1 et ϕ_2 sont deux formules dans GCF_{PS}, nous avons :

- $\phi_1 \wedge \phi_2 \equiv \phi_2 \wedge \phi_1$ et $\phi_1 \vee \phi_2 \equiv \phi_2 \vee \phi_1$.

La définition de la conjonction (resp. disjonction) dans $\mathcal{QCL}+$ diffère légèrement de celle utilisée dans le cadre de la logique $\mathcal{PQCL}$. Si on parle du niveau d'importance, dans le cadre de la logique $\mathcal{QCL}+$, les deux préférences ϕ_1, ϕ_2 ont la même importance dans la mesure où tout ce qui est exprimé est souhaitable, c'est-à-dire que pour les préférences de la forme $\phi_1 \wedge \phi_2$, ϕ_1 est autant préférée ou autant désirable que ϕ_2. Par ailleurs, l'importance s'impose toujours au niveau de ϕ_1 (c-à-d. si $\phi_1 = (a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n)$ alors l'option a_1 est préférée à l'option a_2, etc.) et également au niveau de ϕ_2 (c-à-d. si $\phi_2 = (b_1 \vec{\times} b_2 \vec{\times} \ldots \vec{\times} b_n)$ alors l'option b_1 est préférée à l'option b_2, etc.). Concernant la logique $\mathcal{QCL}+$, toutes les solutions (même celles qui falsifient les préférences) sont plus ou moins possibles, par contre pour $\mathcal{PQCL}$, la solution possible concerne une solution qui satisfait toutes les préférences d'un agent avec un certain degré.

L'optionalité des formules dans le cadre de la logique $\mathcal{QCL}+$ est définie par :

Définition 4.6
Soient ϕ_1 *et* ϕ_2 *deux formules dans* GCF_{PS}.
– *opt(a) = 1 , a est un atome propositionnel.*

- $opt(\phi_1 \vec{\times} \phi_2) = opt(\phi_1) + opt(\phi_2)$.
- $opt(\phi_1 \wedge \phi_2) = max(opt(\phi_1), opt(\phi_2))$.
- $opt(\phi_1 \vee \phi_2) = max(opt(\phi_1), opt(\phi_2))$.
- $opt(\neg(\phi_1)) = opt(\phi_1)$.

L'optionalité de la conjonction (resp. disjonction) de deux formules est le maximum des options que chaque formule présente. Donc, la définition de l'optionalité de la conjonction et la disjonction dans $\mathcal{QCL}+$ est différente de celle utilisée dans la logique $\mathcal{PQCL}$. Par ailleurs, la définition de l'optionalité d'autres opérateurs, en particulier la négation, est identique à celle définie dans la logique $\mathcal{PQCL}$.

4.3.2 Relation d'inférence $\mathcal{QCL}+$

La relation d'inférence dans le cadre de la logique $\mathcal{QCL}+$, notée par $\mid\!\sim^{QCL+}$, est basée sur la fonction de normalisation $\mathcal{N}_{\mathcal{QCL}+}$. La relation d'inférence des formules propositionnelles et formules de choix de base, est la même que celle définie pour la logique $\mathcal{QCL}$ (Définition 2.3). Pour les formules de choix générales, la relation d'inférence est donnée comme suit :

- Appliquer la définition 4.5 pour transformer toute formule de choix générale en une formule de choix de base.
- Appliquer la définition 2.3 pour définir le degré de satisfaction des formules de choix de base obtenues.

Exemple 4.6
Soit $\phi = ((a \vee \neg b) \vec{\times} c) \wedge d$, et $I = \{b,c\}$.
Pour définir le degré de satisfaction associé à la formule ϕ par l'interprétation I, nous utilisons dans un premier temps $\mathcal{N}_{\mathcal{QCL}+}$ pour transformer ϕ qui une formule de choix générale, en une formule de choix de base.
$\mathcal{N}_{\mathcal{QCL}+}(((a \vee \neg b) \vec{\times} c) \wedge d) \equiv \mathcal{N}_{\mathcal{QCL}+}(((a \vee \neg b) \wedge d) \vec{\times} c\ )$
$\equiv ((a \vee \neg b) \wedge d) \vec{\times} c$ (d'après l'item 3-(a) et l' item (1) de la définition 4.5).

La formule obtenue est de choix de base. Nous pouvons donc appliquer la définition 2.3 (Inférence des formules BCF) pour définir le degré de satisfaction de ϕ par l'interprétation I.
Nous avons $I \not\models ((a \vee \neg b) \wedge d)$, $I \models c$, donc $I \mid\!\sim_2^{QCL+} \phi$.

Dans ce qui suit, nous définissons la notion des modèles préférés dans le cadre de la logique $\mathcal{QCL}+$, qui en effet une version légèrement modifiée de la définition 2.8.

Soit K un ensemble de formules propositionnelles qui représentent les connaissances ou les contraintes d'intégrités, et soit T un ensemble de préférences.

Définition 4.7
Soit $I^k(T)$ dénote l'ensemble des formules de T satisfaites par le modèle I avec le degré k. Un modèle I_1 est $K \cup T$-préféré au modèle I_2, noté $I_1 \succ_p I_2$ si il existe un k tel que $\mid I_1^k(T)\mid > \mid I_2^k(T)\mid$ et pour tout $j < k$: $\mid I_1^j(T)\mid = \mid I_2^j(T)\mid$.
I est un $\mathcal{QCL}+$modèle préféré de $K \cup T$ ssi :

1. *I est un modèle de K,*
2. *I est maximum $K \cup T$-préféré au sens de la relation $\succ_p$.*

La différence principale entre la définition des modèles préférés dans le cadre de la logique $\mathcal{QCL}+$ et celle de $\mathcal{QCL}$ (Définition 2.8), concerne l'item (1). Il n'est plus requis que I satisfasse chaque préférence avec un certain degré. Cela est dû au fait que la logique $\mathcal{QCL}+$ est appropriée à la représentation des préférences positives, et la violation de ces préférences ne devrait pas exclure des solutions.

Nous introduisons la définition de la relation d'inférence $\mathcal{QCL}+$ entre les formules de la base $K \cup T$ et une formule propositionnelle, qui est similaire à la définition 2.9, à condition que toute formule de choix générale soit en premier transformée en une formule de choix de base en utilisant $\mathcal{N}_{\mathcal{QCL}+}$.

Définition 4.8
Soient K un ensemble de formules propositionnelles et T un ensemble de préférences représentées par des formules de choix générales. Soit T' un ensemble de préférences obtenues de T par le remplacement de chaque ϕ dans T par $\mathcal{N}_{\mathcal{QCL}+}(\phi)$.
Soit ψ une formule propositionnelle de $PROP_{PS}$.
$K \cup T \mid\!\sim^{QCL+} \psi$ si ψ est satisfaite dans tous les $\mathcal{QCL}+$modèles préférés de $K \cup T'$.

Dans ce qui suit, nous nous basons sur l'exemple 4.6 pour illustrer, en effet, que dans le cadre $\mathcal{QCL}+$, $\neg\neg\phi \equiv \phi$:

Nous avons :
$\mathcal{N}_{\mathcal{QCL}+}(\neg\neg\phi) \equiv \mathcal{N}_{\mathcal{QCL}+}(\neg\neg(((a \vee \neg b)\vec{\times}c) \wedge d))$
$\equiv \mathcal{N}_{\mathcal{QCL}+}(\neg\, \mathcal{N}_{\mathcal{QCL}+}(\neg(((a \vee \neg b)\vec{\times}c) \wedge d)))$
$\equiv \mathcal{N}_{\mathcal{QCL}+}(\neg\, \mathcal{N}_{\mathcal{QCL}+}(\neg\, \mathcal{N}_{\mathcal{QCL}+}\ (((a \vee \neg b)\vec{\times}c) \wedge \mathrm{d})))$
$\equiv \mathcal{N}_{\mathcal{QCL}+}(\neg(\mathcal{N}_{\mathcal{QCL}+}(\neg(((a \vee \neg b) \wedge d)\vec{\times}c))))$
$\equiv \mathcal{N}_{\mathcal{QCL}+}(\neg(\mathcal{N}_{\mathcal{QCL}+}(\neg((a \vee \neg b) \wedge d)\vec{\times}\neg c))))$
$\equiv \mathcal{N}_{\mathcal{QCL}+}(\neg(\neg((a \wedge \neg b) \wedge d)\vec{\times}\neg c)$
$\equiv \mathcal{N}_{\mathcal{QCL}+}(\neg\neg((\mathrm{a} \vee \neg b) \wedge d)\vec{\times}\neg\neg c)$
$\equiv ((\mathrm{a} \vee \neg b) \wedge d)\vec{\times}c$

Donc, $I \mid\!\sim_2^{\mathcal{N}_{\mathcal{QCL}+}} \phi$ et $I \mid\!\sim_2^{\mathcal{N}_{\mathcal{QCL}+}} \neg\neg\phi$. Ces deux formules ont le même degré de satisfaction, il reste à vérifier l'optionalité de chacune des deux formules.

D'un coté, nous avons :

- $\mathrm{opt}(\phi) = \max(\mathrm{opt}((\mathrm{a} \vee \neg b)\vec{\times}c), opt(d))$
  $= \max(\mathrm{opt}((\mathrm{a} \vee \neg b) + opt(c)), opt(d))$
  $= \max(1{+}1,\ 1)$
  $= 2.$

D'un autre coté, nous avons :

- $\mathrm{opt}(\neg\neg\phi) = \mathrm{opt}(\mathcal{N}_{\mathcal{QCL}+}(\neg\neg\phi)))$
  $= \mathrm{opt}((\mathrm{a} \vee \neg b) \wedge d) + opt(c)$
  $= 2$

Les deux formules ont des degrés de satisfaction égaux, et également même optionalité, donc $\neg\neg\phi \equiv \phi$.

L'exemple suivant illustre que contrairement à la logique $\mathcal{QCL}$ qui est incohérente lorsque nous avons considéré les deux bases K et T introduites dans l'exemple 3.1. La logique $\mathcal{QCL}+$ comme $\mathcal{PQCL}$ n'est pas incohérente.

Exemple 4.7
Considérons l'exemple 3.1 où la base de connaissances K contient :

$$\phi_1 = \neg \mathit{EasyJet} \vee \neg \mathit{Air\text{-}France}$$

et la base de préférences T contient les préférences suivantes :

$$\begin{cases} \phi_2 = \neg(\mathit{Air\text{-}France} \vec{\times} \mathit{EasyJet}) \vee \mathit{Formule\text{-}Hotel} \\ \phi_3 = \neg(\mathit{EasyJet} \vec{\times} \mathit{Air\text{-}France}) \vee \neg \mathit{Formule\text{-}Hotel} \\ \phi_4 = \mathit{Air\text{-}France} \vec{\times} \mathit{EasyJet} \end{cases}$$

Pour définir l'ensemble des modèles préférés de $K \cup T$, nous définissons en premier le degré de satisfaction des formules ϕ_1, ϕ_2, ϕ_3 et ϕ_4 pour toute interprétation. Nous utilisons ainsi, $\mathcal{N}_{\mathcal{QCL}+}$ pour transformer les préférences qui sont respectivement représentées par les formules de choix générales ϕ_2 et ϕ_3 en formules de choix de base $\phi_{2'}$ $\phi_{3'}$. Par la suite, nous utilisons la définition 2.3 pour définir les degrés de satisfaction des préférences ϕ_1, $\phi_{2'}$, $\phi_{3'}$, ϕ_4.

Le tableau suivant donne les degré de satisfaction des formules de K et T. Pour toute interprétation, le nombre (1, 2) indique le degré de satisfaction de la formule si elle est satisfaite ou (-) si elle est falsifiée.

Air-France	*EasyJet*	*Formule-Hotel*	ϕ_1	$\phi_{2'}$	$\phi_{3'}$	ϕ_4
F	F	F	1	1	1	-
F	F	T	1	1	1	-
F	T	F	1	1	1	2
F	T	T	1	1	2	2
T	F	F	1	2	1	1
T	**F**	**T**	**1**	**1**	**1**	**1**
T	T	F	-	-	1	1
T	T	T	-	1	-	1

TAB. 4.2: Les modèles de la base $T \cup K$ dans le cadre de la relation d'inférence $\mathcal{QCL}+$

Les six premières interprétations satisfont la condition (1) de la définition 4.7 des modèles préférés. Cependant, uniquement la sixième interprétation (ligne en gras) I={*Air-France, Formule-Hotel*} qui est un modèle préféré.

Exemple 4.8
Considérons cet exemple, où nous nous limitons à l'ensemble de préférences suivantes :

1. "Un voyage avec deux escales est préféré à celui avec 3 escales qui est aussi préféré à un voyage ayant strictement plus de 3 escales"
2. "Les personnes qui préfèrent AF à (KLM ou NW), préfèrent Paris comme escale"
3. "Les personnes qui préfèrent KLM à (AF ou NW) préfèrent Amsterdam comme escale"

4. "Les personnes qui préfèrent NW à (AF ou KLM) préfèrent Detroit comme escale"

En plus, supposons que nous avons d'autres préférences concernant un agent particulier :

- "Il préfère AF à (KLM ou NW)"
- "Les voyages proposés dont le nombre d'escales est plus que 4 ou le prix est supérieur à 1500 euros sont inacceptables".

Pour coder ces préférences, nous utilisons les variables suivantes :

- AF (resp. NW, KLM) pour coder le fait que AF concerne la compagnie aérienne préférée (resp. NW, KLM),
- ≥ 2 escales (resp. ≥ 3, ≥ 4), pour coder le fait que le voyage contient plus de 2 escales (resp. plus de 3 escales, plus de 4 escales),
- Paris (resp. Amsterdam, Detroit, autre) : pour coder le fait que le voyage contient Paris (resp. Amsterdam, Detroit, autre) comme un pays d'escale,
- $\leq$ 1500 euros (le billet d'avion coûte moins de 1500 euros).

Étant données les différentes variables, notre base de connaissances K contient :

$$K = \neg\ (\geq 4 \text{ escales}) \wedge (\leq 1500 \text{ euros}) \qquad \textbf{(1)}$$

et la base de préférences T contient :

$$T = \begin{cases} (\geq 2 \quad escales) \vec{\times} (\geq 3 \quad escales) \vec{\times} (\geq 4 \quad escales) & (\mathbf{2}) \\ AF \vec{\times} KLM \vec{\times} NW \qquad (\mathbf{3}) \\ AF \vec{\times} (KLM \vee NW) \Rightarrow Paris \qquad (\mathbf{4}) \\ KLM \vec{\times} (AF \vee NW) \Rightarrow Amsterdam \qquad (\mathbf{5}) \\ NW \vec{\times} (AF \vee KLM) \Rightarrow Detroit \qquad (\mathbf{6}) \end{cases}$$

A partir de ces informations, la solution attendue de la relation d'inférence $\mathcal{QCL}+$ est {AF,$\geq$ 2 escales,$\leq$ 1500 euros} qui est la solution préférée.

Ainsi, notre logique peut être utile, dans la mesure où : il est d'abord utile de représenter les préférences simples des agents (ou profils), mais aussi des règles complexes génériques qui impliquent des préférences. Notre logique est également utile pour déterminer des solutions qui seront présentées aux utilisateurs. Enfin, la notion du degré de satisfaction est importante pour les algorithmes de recherche, et notamment pour la taille de l'espace de recherche de solutions.

4.4 Propriétés de nos logiques

Cette section introduit quelques propriétés des trois alternatives que nous avons proposées ($\mathcal{MQCL}$ qui est introduite dans le chapitre précédent et $\mathcal{PQCL}$, $\mathcal{QCL}+$ présentées dans ce chapitre). Le choix de la logique à utiliser dépend de l'application considérée. Si nous traitons des préférences prioritaires, où les choix des agents sont associés des niveaux d'importance différents, alors la logique $\mathcal{PQCL}$ est la plus appropriée, par contre lorsque les préférences sont positives, où les agents expriment tout ce qui considèrent souhaitables pour eux, alors il est préférable d'appliquer la logique $\mathcal{QCL}+$. Quant à la logique $\mathcal{MQCL}$, elle peut être appliquée pour représenter différents types de préférences autres que prioritaires et positives. En effet, la différence principale se situe au niveau de la sémantique de la conjonction et de la disjonction de chaque logique. Les principales caractéristiques des ces logiques sont les suivantes :

La négation

Concernant les logiques $\mathcal{PQCL}$ et $\mathcal{QCL}+$, notre définition de la négation retrouve les propriétés propositionnelles de la négation telles que par exemple les lois de De Morgan ou la double négation comme nous l'illustrons dans la proposition suivante :

Proposition 4.2
La double négation d'une formule dans GCF_{PS} lorsque nous considérons $\mathcal{PQCL}$ ou $\mathcal{QCL}+$ est équivalente à la même formule, c'est-à-dire $\neg\neg(\phi) \equiv \phi$.
La négation est décomposable par rapport à la conjonction et la disjonction lorsque nous considérons les logiques $\mathcal{PQCL}$ et $\mathcal{QCL}+$. Plus précisément :
$\neg(\phi_1 \wedge \phi_2) \equiv \neg\phi_1 \vee \neg\phi_2$, *et* $\neg(\phi_1 \vee \phi_2) \equiv \neg\phi_1 \wedge \neg\phi_2$.

La preuve de cette proposition est immédiate, elle découle de la définition 4.4 pour $\mathcal{PQCL}$ et la définition 4.5 pour $\mathcal{QCL}+$.

Concernant la logique $\mathcal{MQCL}$, la double négation d'une formule donne aussi la même formule, mais la négation n'est pas décomposable par rapport à la conjonction et la disjonction (voir section 3.4.4).

La relation d'inférence

Premièrement, il est à noter qu'une formule propositionnelle peut être équivalente à une formule de choix de base bien particulière. Nous avons :

Proposition 4.3
Soit ϕ une formule propositionnelle dans $PROP_{PS}$ et soit I une interprétation. Alors,
$I \models_1 \phi$ *ssi* $I \models_1 \phi \vec{\times} \bot$ *ssi* $I \models \phi$.

La preuve de cette proposition est immédiate. Si $I \models_1 \phi$, alors I satisfait la première option de $\phi \vec{\times} \bot$, donc $I \models_1 \phi \vec{\times} \bot$. Notons que lorsque la formule $\phi \vec{\times} \bot$ est satisfaite par I, alors son degré de satisfaction est unique et égal à 1. $I \models_1 \phi \vec{\times} \bot$ signifie que $I \models \phi$, donc, d'après la définition 2.3, $I \models_1 \phi$.

Une autre propriété intéressante concerne le fait que le degré de satisfaction de toute formule dans le cadre des trois logiques est unique. Plus précisément :
Soit ϕ dans GCF_{PS}. $\forall I \in \Omega$,
si $I|\sim_i^{PQCL}\phi$ et $I|\sim_j^{PQCL}\phi$ (resp. $I|\sim_i^{QCL+}\phi$ et $I|\sim_j^{QCL+}\phi$, $I|\sim_i^{MQCL}\phi$ et $I|\sim_j^{MQCL}\phi$) alors $i = j$.

Les logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$ sont considérées comme des extensions de la logique propositionnelle. En particulier, la conjonction (resp. disjonction) dans le cadre de $\mathcal{PQCL}$ et $\mathcal{QCL}+$ se comportent comme la conjonction (resp. disjonction) propositionnelle.

Lorsque nous utilisons la logique $\mathcal{PQCL}$, $\mathcal{QCL}+$ ou $\mathcal{MQCL}$ pour exprimer les connaissances des agents, une formule propositionnelle de $PROP_{PS}$ si elle est satisfaite, son degré de satisfaction est toujours égal à 1. Plus précisément, nous avons :

Proposition 4.4
Soit ϕ une formule propositionnelle, et I une interprétation. Alors :
$I|\sim_1^{PQCL}\phi$ *ssi* $I|\sim_1^{QCL+}\phi$ *ssi* $I|\sim_1^{MQCL}\phi$ *ssi* $I \models \phi$.

Preuve de la proposition 4.4

Pour montrer la proposition 4.4, il suffit de réécrire la définition 4.2 (dans le cas de la relation d'inférence $\mathcal{PQCL}$) lorsque nous nous limitons aux formules propositionnelles, et nous montrons que la relation d'inférence obtenue se rejoigne à l'inférence propositionnelle. En premier, il est facile de montrer que l'optionalité de toute formule propositionnelle est égale à 1. La preuve peut se faire par récurrence. En effet, dans la définition 4.1, l'optionalité d'un atome a est égale à 1. Supposons que ϕ_1 et ϕ_2 sont des formules propositionnelles et leur optionalité est égale à 1. Alors, $opt(\neg\phi_1) = opt(\neg\phi_2) = 1$, $opt(\phi_1 \wedge \phi_2) = opt(\phi_1) \times opt(\phi_2) = 1$ et $opt(\phi_1 \vee \phi_2) = opt(\phi_1) \times opt(\phi_2) = 1$.
Notons que l'item (3) et l'item (9) de la définition 4.2 peuvent être supprimés du fait que nous ne traitons pas des préférences, c'est-à-dire avec de la disjonction ordonnée $\vec{\times}$.
De plus, si une formule propositionnelle est satisfaite, alors le degré de satisfaction est égal à 1. Nous pouvons aussi vérifier cela par récurrence. Premièrement, les atomes propositionnels peuvent être satisfaits avec uniquement le degré 1 en utilisant l'item (1) de la définition 4.2. Ainsi, si ϕ et ψ sont satisfaites avec le degré 1, alors en utilisant l'item (5) de la définition 4.2, $\phi \wedge \psi$ est satisfaite avec le degré $k = (1-1) \times opt(\psi) + 1 = 1$. Notons que dans ce cas, le degré de satisfaction de $\psi \wedge \phi$ est aussi égal à 1. Lorsque nous considérons uniquement les formules propositionnelles, notre conjonction est commutative. D'une façon similaire, si ϕ ou ψ est satisfaite avec le degré 1 alors par l'application de l'item(4-1), $\phi \vee \psi$ est satisfaite avec le degré $k = 1$. Ainsi, le degré de satisfaction de $\psi \vee \phi$ est aussi égal à 1, et donc notre disjonction est aussi commutative. La négation d'une formule propositionnelle (si elle est satisfaite) peut être satisfaite avec uniquement le degré 1.

Comme toute formule propositionnelle, si elle est satisfaite, son degré de satisfaction est égal à 1, alors les items (4-b) et (4-c) peuvent être supprimés (puisque les conditions de leur application sur des formules propositionnelles sont toujours fausses).
Étant donnés les faits ci-dessus, la définition 4.2 pour les formules propositionnelles devient comme suit :
Soit ϕ_1, ϕ_2 deux formules propositionnelles. Soit I une interprétation.

1. $I \mid\sim_1^{PQCL}$ a *ssi* a $\in I$ *(pour les atomes propositionnels a).*

2. $I \mid\sim_1^{PQCL} \neg$a *ssi* $\neg$ a $\in I$ *(pour les atomes propositionnels a).*

4. $I \mid\sim_1^{PQCL} \phi_1 \vee \phi_2$ *ssi* $I \mid\sim_1^{PQCL} \phi_1$ *ou* $I \mid\sim_1^{PQCL} \phi_2$,

5. $I \mid\sim_1^{PQCL} \phi_1 \wedge \phi_2$ *ssi* $I \mid\sim_1^{PQCL} \phi_1$ *et* $I \mid\sim_1^{PQCL} \phi_2$

6. $I \mid\sim_1^{PQCL} \neg(\phi_1 \vee \phi_2)$ *ssi* $I \mid\sim_1^{PQCL} \neg\phi_1 \wedge \neg\phi_2$.

7. $I \mid\sim_1^{PQCL} \neg(\phi_1 \wedge \phi_2)$ *ssi* $I \mid\sim_1^{PQCL} \neg\phi_1 \vee \neg\phi_2$.

9. $I \mid\sim_1^{PQCL} \neg(\neg\phi_1)$ *ssi* $I \mid\sim_1^{PQCL} \phi_1$.

Il est clair que cela correspond à l'inférence propositionnelle, c'est-à-dire que pour chaque formule propositionnelle ϕ et une interprétation I, nous avons : $I \mid\sim_1^{PQCL} \phi$ *ssi* $I \models \phi$. Concernant $\mathcal{QCL}+$

et $\mathcal{MQCL}$, la preuve est immédiate, elle est basée directement sur la fonction de normalisation de chaque logique. □

La relation d'inférence au niveau de BCF_{PS} : une formule de choix de base de BCF_{PS} correspond à la disjonction ordonnée de plusieurs formules propositionnelles. Cela signifie que les différentes alternatives concernées sont ordonnées de la plus préférée à la moins préférée. Comme les préférences représentées par les formules BCF sont d'une structure simple, alors ces préférences constituent un ordre unique dans le cadre des logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$. Plus précisément, nous avons :

Proposition 4.5
Soient $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ une formule de choix de base, où les a_i sont des formules propositionnelles, et I une interprétation, et soient $\mid\!\sim^{PQCL}$, $\mid\!\sim^{QCL+}$ et $\mid\!\sim^{MQCL}$ trois relations qui représentent respectivement les relations d'inférence pour $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mid\!\sim^{MQCL}$, alors :
$I \mid\!\sim_k^{PQCL} \phi$ ssi $I \mid\!\sim_k^{QCL+} \phi$ ssi $I \mid\!\sim_k^{MQCL} \phi$ ssi $I \models a_1 \vee a_2 \vee \ldots \vee a_n$ et $k = min\{j \mid I \models a_j\}$.

La preuve de cette proposition est immédiate. Concernant les formules de choix de base, la relation d'inférence définie dans la définition 2.3 pour la logique $\mathcal{QCL}$ reste valable dans le cadre de $\mathcal{QCL}+$, $\mathcal{PQCL}$ et $\mathcal{MQCL}$, de plus, par exemple, étant donnée une interprétation, la relation d'inférence $\mathcal{PQCL}$ (item (3)) de la définition 4.2 est identique à celle donnée dans la définition 2.3. □

Monotonie : les logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$ sont non-monotones. A titre d'exemple, considérons T = {a $\vec{\times}$b}. Nous avons trois modèles {a}, {a, b}, {b} avec respectivement les degrés de satisfaction 1, 1, 2. Cela signifie que les modèles {a} et {a, b} sont maximum-préférés, et nous avons :
a $\vec{\times}$ b $\mid\!\sim^{PQCL}$ a (resp. a $\vec{\times}$ b $\mid\!\sim^{QCL+}$ a, et a $\vec{\times}$ b $\mid\!\sim^{MQCL}$ a).

Si nous rajoutons l'information ¬a, alors l'unique modèle qui est aussi l'unique modèle préféré, est {b}. Nous obtenons donc
{a $\vec{\times}$b, ¬a} $\mid\!\sim^{PQCL}$ ¬a (resp. {a $\vec{\times}$b, ¬a} $\mid\!\sim^{QCL+}$ ¬a, {a $\vec{\times}$b, ¬a} $\mid\!\sim^{MQCL}$ ¬a). Donc, a n'est pas conclue.

4.5 Relations avec la logique possibiliste

4.5.1 La logique possibiliste et les formules de choix de base

Le langage des formules de choix de base a une forte relation avec la théorie des possibilités. Dans [Benferhat *et al.* 2004], les auteurs ont montré que toute formule de choix de base peut être représentée comme une base de connaissance possibiliste. Dans cette section, nous rappelons et étendons ces relations dans le cadre de nos logiques.

Comme nous l'avons vu dans le chapitre 1, la logique possibiliste est vue comme un modèle simple et naturel pour le raisonnement et le traitement des situations d'incertitude. Les informations incertaines sont représentées par des distributions de possibilités. Une distribution de possibilités notée par π, est une fonction de l'ensemble des solutions ou interprétations vers l'intervalle [0,1]. Par convention, $\pi(I) = 1$ signifie que I est la solution la plus normale (ou préférée). $\pi(I) = 0$

signifie que I est impossible. $\pi(I) \geq \pi(I')$ signifie que I est au moins autant préférée que I'. Récemment, plusieurs auteurs se sont intéressés [Benferhat & Kaci 2003, Dubois *et al.* 2000] à une représentation compacte des informations en logique possibiliste, appelée *base de possibilité garantie* ou simplement Δ-base. Cette représentation est basée sur la notion de *mesure de possibilité garantie*, comme nous l'avons définie dans le chapitre 1 (voir Définition 1.2).

Une base de possibilité garantie est composée d'un ensemble de formules pondérées de la forme $[\phi_i, \alpha_i]$. La paire $[\phi_i, \alpha_i]$ signifie que toute formule ϕ_i satisfaite par un modèle, est associée d'un degré au moins égal à α_i, c'est-à-dire :

$$\Delta(\phi_i) \geq \alpha_i$$
ou
$$\forall I, I \models \phi_i, \pi(I) \geq \alpha_i.$$

Définition 4.9
Toute base de possibilité garantie Δ induit une unique distribution de possibilités π tel que pour toute interprétation I,

$$\pi(I) = \begin{cases} 0 \quad si \quad I \quad falsifie \quad toutes \quad les \quad formules \quad de \quad \Delta, \\ max\{\alpha_i : [\phi_i, \alpha_i] \in \Delta, I \models \phi_i\} \quad sinon \end{cases}$$

La proposition suivante indique qu'une formule propositionnelle peut être représentée par une simple base de possibilité garantie, plus précisément :

Proposition 4.6
Soit ϕ une formule propositionnelle. Soit $\Delta_\phi = \{[\phi, 1]\}$ une base de possibilité garantie. Alors, pour chaque interprétation $I \in \Omega$, $I \models \phi$ ssi $I \models_1 \phi$ ssi $\pi(I) = 1$ et $I \not\models \phi$ ssi $\pi(I) = 0$.

La preuve est immédiate. Soit $\Delta_\phi = \{[\phi, 1]\}$ une simple base de garantie possibiliste. Cela signifie que Δ_ϕ induit une distribution de possibilités π où, pour toute interprétation I, nous avons :

- $\pi(I) = 0$ si I falsifie ϕ, cela signifie que $I \not\models \phi$,
- $\pi(I) = 1$ si I satisfait ϕ, cela signifie que $I \models \phi$. □

La proposition suivante introduit la relation entre la logique $\mathcal{PQCL}$ (resp. $\mathcal{QCL}+$, $\mathcal{MQCL}$) et la logique possibiliste lorsque nous avons uniquement une formule de choix de base.

Proposition 4.7
[Brewka et al. *2004] Soit $\phi = a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$ une formule de choix de base. Soit $\alpha_i = \varepsilon^i$ pour i=1,...,n, tel que $0 < \varepsilon < 1$. Soit $\Delta = \{[a_1, \alpha_1], ..., [a_n, \alpha_n]\}$ une Δ-base associée à ϕ. Alors $\forall I$, $I \models_k \phi$ ssi $\pi(I) = \alpha_k$ et $I \not\models \phi$ ssi $\pi(I) = 0$.*

La preuve de cette proposition est immédiate. En effet, d'un coté, s'il n'existe pas un i tel que $I \models A_i$, alors selon la définition 4.9, nous avons $\pi(I) = 0$. D'un autre coté, selon la définition 2.3, nous avons $I \not\models \phi$.
Supposons que I satisfait certaines options a_i alors, $\pi(I) = \alpha_k$ signifie que $I \models a_k$, et pour $i = 1,...,k\text{-}1$, nous avons $I \not\models a_i$, et cela est exactement équivalent à $I \models_k \phi$. Donc, toute formule de choix de base peut être représentée par Δ-*base*. □

4.5.2 Une définition alternative des modèles préférés

Dans cette section, nous introduisons une autre définition des modèles préférés en se basant sur la fusion des distributions de possibilités. La logique $\mathcal{PQCL}$ (resp. $\mathcal{QCL}+$, $\mathcal{MQCL}$) se base sur des critères lexicographiques pour ordonner les modèles préférés. La définition 4.10 donne une autre alternative basée sur des opérateurs maximum de base.

Définition 4.10
Soit $T = \{\phi_1, ..., \phi_n\}$ un ensemble de formules de GCF_{PS}. Un modèle I_1 de T est max-préféré de T ssi il n'existe pas un modèle I_2 de T tel que $max(k^{I_2}(\phi_1), ..., k^{I_2}(\phi_n)) < max(k^{I_1}(\phi_1), ..., k^{I_1}(\phi_n))$, où $k^{I_i}(\phi_j)$ représente le degré de satisfaction de la formule ϕ_j par le modèle I_i.
La relation d'inférence induite par cette relation de préférence est notée par $\mid\sim_{max}^{PQCL}$ (resp. $\mid\sim_{max}^{QCL+}$, $\mid\sim_{max}^{MQCL}$).

Dans ce qui suit, nous présentons les relations qui peuvent être établies entre la logique possibiliste et la relation $\mid\sim_{max}^{PQCL}$ (resp. $\mid\sim_{max}^{QCL+}$, $\mid\sim_{max}^{MQCL}$). L'idée principale consiste à obtenir la base possibiliste associée à la théorie $\mid\sim_{max}^{PQCL}$ (resp. $\mid\sim_{max}^{QCL+}$, $\mid\sim_{max}^{MQCL}$). Pour ce faire, il suffit de fusionner les bases de possibilités garanties associées pour chaque formule $\mathcal{PQCL}$ (resp. $\mathcal{QCL}+$, $\mathcal{MQCL}$) dans T. La fusion de plusieurs bases possibilistes garanties est définie dans la définition 4.11.

Définition 4.11
Soit $\Delta_1, ..., \Delta_n$, n bases possibilistes garanties, le résultat de la fusion de ces bases, noté par $\Delta_\oplus$, est défini par $\Delta_\oplus = \{min\lfloor\phi_{1i} \wedge ... \wedge \phi_{nk}, min(\alpha_{1i}, ..., \alpha_{nk})\rfloor \mid \lfloor\phi_{1i}, \alpha_{1i}\rfloor \in \Delta_1, ..., \lfloor\phi_{nk}, \alpha_{nk}\rfloor \in \Delta_n\}$.

Nous pouvons vérifier que la distribution de possibilités associée à $\Delta_\oplus$ définie précédemment correspond au plus petit degré de possibilité associé aux bases de garanties possibilistes :

Proposition 4.8
Soient $\Delta_1, ..., \Delta_n$ un ensemble de bases possibilsites garanties (Δ-bases). Soient $\pi_1, ..., \pi_n$ des distributions de possibilités associées respectivement à chaque Δ-base. $\Delta_\oplus$ correspond à la base obtenue de la fusion de Δ-bases donnée dans la définition 4.11. Alors $\forall I, \pi_{\Delta_\oplus}(I) = min(\pi_1(I), ..., \pi_n(I))$.

Preuve de la proposition 4.8 Nous allons montrer seulement le cas où deux Δ-bases sont fusionnées. Le cas général se démontre d'une manière similaire.
Rappelons que : $\pi_{\Delta_\oplus} = \{[\phi_i \wedge \psi_j, min(\alpha_i, \beta_j)] : [\phi_i, \alpha_i] \in \Delta_1$ et $[\psi_j, \beta_j] \in \Delta_2\}$.
Notons en premier, que si pour une interprétation I, $\pi_1(I) = 0$ ou $\pi_2(I) = 0$ alors cela signifie que I falsifie toutes les formules propositionnelles de Δ_1, ou I falsifie toutes les formules propositionnelles de Δ_2. Donc, I falsifie aussi toutes les formules propositionnelles de $\Delta_\oplus$. Ainsi, $\pi_{\Delta_\oplus} = 0 = min(\pi_1(I), ..., \pi_n(I))$.
Maintenant, nous supposons que I satisfait au moins une formule propositionnelle de Δ_1 et satisfait au moins une formule propositionnelle de Δ_2. Alors $\pi_{\Delta_\oplus}$ est calculée comme suit :

$$\begin{aligned}\pi_{\Delta_\oplus}(I) &= \max\{min(\alpha_i, \beta_j) : I \models \phi_i \wedge \psi_j \text{ et } [\phi_i \wedge \psi_j, min(\alpha_i, \beta_j)] \in \Delta_\oplus\} \\ &= max\{min(\alpha_i, \beta_j) : I \models \phi_i \wedge \psi_j \text{ et } [\phi_i, \alpha_i] \in \Delta_1, [\psi_j, \beta_j)] \in \Delta_2\} \\ &= max\{min(\alpha_i, \beta_j) : I \models \phi_i, [\phi_i, \alpha_i] \in \Delta_1, \text{ et } I \models \psi_j, [\psi_j, \beta_j)] \in \Delta_2\}\end{aligned}$$

lorsque α_i et β_j sont maximales alors $min(\alpha_i, \beta_j)$ est aussi maximale.
Donc, $\pi_{\Delta_\oplus}(I) = min(max\ \{\alpha_i : I \models \phi_i, [\phi_i, \alpha_i] \in \Delta_1\}, max\ \{\ \beta_j, I \models \psi_j, [\psi_j, \beta_j] \in \Delta_2\}) =$

$min(\pi_1(I), \pi_2(I))$.

Comme corollaire des deux propositions 4.7 et 4.8, il est possible de coder les théories $\mathcal{PQCL}^{max}$ (resp. $\mathcal{QCL}+^{max}$, $\mathcal{MQCL}^{max}$) dans un cadre possibiliste, comme le montre la proposition suivante :

Proposition 4.9
Soit $T = \{\phi_1, ..., \phi_n\}$ une théorie $\mathcal{PQCL}$ (resp. $\mathcal{QCL}+$, $\mathcal{MQCL}$), et supposons que la base de connaissance K est vide. Soit Δ_{ϕ_i} la base de possibilité garantie Δ-base associée à ϕ_i utilisant la proposition 4.7 et soit $\Delta_\oplus$ le résultat de fusion des bases Δ_{ϕ_i} utilisant la définition 4.11. Alors, pour toute interprétations I et I', I est max-préféré à I' (Définition 4.10) ssi $\pi_{\Delta_\oplus}(I) > \pi_{\Delta_\oplus}(I')$.

La preuve de la proposition 4.9 (qui est un corollaire) est immédiate, elle découle des propositions 4.7 et 4.8.

4.5.3 La négation dans les préférences

Dans le cadre des logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$, nous avons donné une seule définition de la négation. Dans notre définition, si nous avons des préférences représentées par $a_1 \vec{\times} a_2 \vec{\times} \ldots \vec{\times} a_n$, a_1 ne doit pas être considérée comme une première option, a_2 ne pas être considérée comme une deuxième option, etc. En se basant sur les résultats des sections précédentes, il est possible de donner une autre définition de la négation qui est utilisée dans la théorie des possibilités. En fait, dans la théorie des possibilités, le concept du complément d'une distribution de possibilités π est naturellement défini par la fonction de reversement notée Ne. Cette fonction de renversement indique que pour chaque interprétation I, I', nous avons :

$$\pi(I) > \pi(I') \text{ ssi } Ne(\pi(I)) > Ne(\pi(I'))$$

Un exemple typique d'une fonction de renversement est définie par :

$$\forall I \in \Omega \ \pi'(I) = 1 - \pi(I)$$

Toutefois, la négation d'une base possibiliste n'a pas été considérée, et n'a pas été définie auparavant. La négation possibiliste est définie par :

Définition 4.12
Soit $\phi = a_1 \vec{\times} a_2 ... \vec{\times} a_n$. La définition possibiliste de la négation est : $\neg\phi = \neg a_1 \wedge \neg a_2 \wedge ... \wedge \neg a_n \vec{\times} a_n \wedge \neg a_{n-1} \wedge ... \wedge \neg a_1 \vec{\times} a_{n-1} \wedge \neg a_{n-2} \wedge ... \wedge \neg a_1 \vec{\times} .... \vec{\times} a_2 \wedge \neg a_1 \vec{\times} a_1$

Cette définition signifie que la meilleure option (la plus préférée) dans $\neg\phi$ code la situation où toutes les options (a_i) dans ϕ sont fausses. Le seconde option dans $\neg\phi$ code la cas où la dernière option dans ϕ (c-à-d. a_n) est vraie, et toutes les autres options sont fausses. La troisième option dans $\neg\phi$ concerne la seconde mauvaise option dans ϕ (c-à-d. a_{n-1}) lorsqu'elle est vraie et toutes les premières options $a_2 ... a_{n-2}$ sont fausses. Et finalement, la mauvaise option (la moins préférée) dans $\neg\phi$ correspond à la première option de ϕ.

Exemple 4.9
Supposons que les préférences d'un agent concernant un dîner simple sont les suivantes : Il préfère un plat à base de poisson qu'un plat contenant de la viande. Ces préférences sont représentées par la formule $\phi = poisson \vec{\times} viande$.
$\neg\phi = (\neg poisson \wedge \neg viande) \vec{\times} (viande \wedge \neg poisson) \vec{\times} poisson$

Nous suggérons que cette définition de la négation est plus forte que celle que nous avons proposée pour les logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$. A titre d'exemple, lorsque nous considérons la définition proposée dans le cadre de nos logiques, nous exigeons que la première option de $\neg\phi$ ne doit pas être la première option de ϕ. Alors que dans la définition 4.12, il n'est pas seulement exigé que la première option ne doit pas être la première option de ϕ, mais elle ne doit pas être aussi la seconde option de ϕ, ni la troisième,... ni d'ailleurs la dernière option de ϕ.

Notons aussi que la définition de la négation possibiliste étend la définition utilisée dans la logique $\mathcal{QCL}$ dans le sens où la première option de ϕ dans la définition 4.12 (négation possibiliste) représente simplement la définition complète de la définition de la négation donnée dans le cadre de la logique $\mathcal{QCL}$. Cependant, la définition de la négation possibiliste contient beaucoup plus d'options.

La proposition suivante montre que la définition 4.12 code réellement la négation utilisée dans le cadre de la théorie des possibilités :

Proposition 4.10
soit Δ_ϕ une Δ-base garantie possibiliste associée à $\phi = a_1 \vec{\times} ... \vec{\times} a_n$ (utilisant la proposition 4.7). Soit également $\Delta_{\neg\phi}$ une base de garantie possibiliste associée à $\neg\phi$ (où $\neg\phi$ est donnée dans la définition 4.12). Alors :
$\forall I, I' \in \Omega$ $\pi_{\Delta_\phi(I)} > \pi_{\Delta_\phi(I')}$ *ssi* $\pi_{\Delta_{\neg\phi}(I')} > \pi_{\Delta_{\neg\phi}(I)}$.

Preuve de la proposition 4.10
Soit $\phi = a_1 \vec{\times} a_2 ... \vec{\times} a_n$. Rappelons que la définition de la négation possibiliste est :
$\neg\phi = \neg a_1 \wedge \neg a_2 \wedge ... \wedge \neg a_n \vec{\times} a_n \wedge \neg a_{n-1} \wedge ... \wedge \neg a_1 \vec{\times} a_{n-1} \wedge \neg a_{n-2} \wedge ... \wedge \neg a_1 \vec{\times} .... \vec{\times} a_2 \wedge \neg a_1 \vec{\times} a_1$.
Notons que toutes les options dans $\neg\phi$ sont mutuellement exclusives. Donc, si une interprétation I donnée satisfait une option de $\neg\phi$, alors elle falsifie nécessairement les autres options.
Nous utilisons la proposition 4.7 pour donner la preuve de cette proposition. Soit I et I' deux interprétations. Supposons que $\pi(I) > \pi(I')$. Donc, nous distinguons deux cas :

- $\pi_{\Delta_\phi}(I') = 0$, signifie que I' falsifie toutes les options a_i, donc $I' \models_1 \neg\phi$. Cela signifie aussi que $\pi_{\Delta_\phi}(I) > 0$, qui signifie qu'il existe j tel que $I \models a_j$, donc, $I \not\models_1 \neg\phi$. En se basant sur la proposition 4.7, nous avons : $\pi_{\Delta_{\neg\phi}}(I') > \pi_{\Delta_{\neg\phi}}(I)$.

- $\pi_{\Delta_\phi}(I) > \pi_{\Delta_\phi}(I') > 0$, signifie que les deux interprétations I et I' satisfont certaines options a_i. Plus précisément, $\pi_{\Delta_\phi}(I) > \pi_{\Delta_\phi}(I') > 0$ implique qu'il existe j et k, tel que $I \models_j \phi$ et $I' \models_k \phi$ et $j < k$ (selon la proposition 4.7 où $\pi_{\Delta_\phi}(I) > \pi_{\Delta_\phi}(I') > 0$). Cela signifie que $I \models \neg a_1 \wedge .. \wedge \neg a_{j-1} \wedge a_j$ et $I' \models \neg a_1 \wedge .. \wedge \neg a_{k-1} \wedge a_k$. Donc $I \models_{n+2-j} \neg\phi$ et $I \models_{n+2-k} \neg\phi$. Comme $j < k$, alors $n + 2 - j > n + 2 - k$, donc $\pi_{\Delta_{\neg\phi}}(I) < \pi_{\Delta_{\neg\phi}}(I')$.

4.6 Conclusion

Nous avons présenté dans ce chapitre deux logiques pour la représentation des préférences qui surmontent les limites et les restrictions de la logique $\mathcal{QCL}$. Ces logiques ($\mathcal{PQCL}$ et $\mathcal{QCL}+$) sont caractérisées par une nouvelle définition de la négation des préférences qui est la même que la logique $\mathcal{MQCL}$ présentée dans le chapitre précédent. Elles proposent également des nouvelles définitions de la conjonction et de la disjonction (prioritaires et positives).

Comme nous l'avons vu dans le premier chapitre, le problème de la représentation des préférences a fait l'objet d'un grand intérêt par de nombreux chercheurs en intelligence artificielle et également par des chercheurs d'autres domaines telles que l'économie, la théorie de choix sociale, la philosophie, etc. Dans la majorité de ces travaux, nous retrouvons certaines relations avec la logique $\mathcal{QCL}$ présentée dans [Brewka *et al.* 2004], $\mathcal{MQCL}$ présentée dans le chapitre 2 et les logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ que nous avons présentées tout au long de ce chapitre. Cependant, un petit nombre parmi les travaux existants permettent de traiter la négation dans les préférences dans le sens où normalement, un agent n'aura pas seulement la possibilité d'exprimer ce qu'il préfère avoir, mais également ce qu'il ne préfère pas avoir. De plus, nos logiques offrent plusieurs avantages par rapport à la majorité de ces travaux. Par exemple, concernant la représentation des préférences dans un cadre graphique. Les CP-nets, sont parmi les modèles graphiques les plus utilisés dans de nombreux travaux. Cependant, ils permettent uniquement de représenter des préférences conditionnelles simples. Nos logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$ offrent plusieurs avantages par rapport à CP-net. A titre d'exemple, CP-net ne peut pas représenter les préférences de la forme : *"les personnes qui préfèrent Air-France à KLM, préfèrent voyager en première classe"*. En fait, les règles conditionnelles ne sont pas réellement traitées par les CP-nets. En effet, dans les CP-nets, les préférences qui sont représentées à base des contextes, sont codées par des conjonctions. Une préférence de la forme *"dans le contexte de Air France, les voyageurs préfèrent le siège fenêtre au siège couloir"* est en effet traitée comme une préférence de la forme : les voyageurs préfèrent *"Air France et fenêtre"* à *"Air France et couloir"*.
Le second avantage de nos logiques par rapport aux CP-nets, est qu'elles permettent une représentation compacte des préférences. Par exemple, supposons qu'une agence de voyage s'intéresse à la catégorie des hôtels (de 2* à 5*) qui peuvent être proposés à ces clients. La proposition de la catégorie d'hôtel dépend des compagnies aériennes (AF, KLM, NW, etc.), la classe de voyage (première classe, économique, affaire, etc.), sexe (féminin, masculin), etc. Pour représenter de telles préférences, les CP-nets procèdent par exemple à fournir les préférences sur les catégories d'hôtels, dans le contexte de "compagnie aérienne, classe de voyage, sexe", etc. Il est clair, qu'il n'est pas facile de fournir ce type d'informations si les domaines des variables sont larges. Le problème est encore plus délicat lorsque pour certains cas, un agent ne dispose d'aucune information concernant les préférence sur la catégorie de l'hôtel.

De plus, comme nous allons le constater dans la suite de la thèse, les logiques $\mathcal{PQCL}$, $\mathcal{QCL}+$ et $\mathcal{MQCL}$ peuvent être également utilisées et appliquées dans d'autres problèmes sensibles et préoccupants. Il s'agit par exemple du problème de la corrélation d'alertes où la logique $\mathcal{MQCL}$ est largement suffisante du fait que les solutions que nous voulons apporter ne nécessitent pas des préférences prioritaires et positives.

Deuxième partie

Application des modèles graphiques et logiques des préférences à la détection d'intrusions et à la corrélation d'alertes

Introduction à la partie II

Les systèmes de détection d'intrusions (IDSs) sont des outils importants qui visent à détecter toute activité malveillante et suspecte au sein d'un système d'informations [Anderson 1980]. Cependant, ces systèmes ne sont pas totalement satisfaisants pour diverses raisons :

- Ces IDSs arrivent à détecter uniquement les attaques connues. Toutes les attaques dont les signatures ne sont pas enregistrées dans la base de signatures par exemple, ne seront pas détectées. Cela est certainement dû au manque d'informations permettant de repérer ces attaques.
- Ils génèrent de nombreuses quantités d'alertes, dont une majorité sont des fausses alertes. Ainsi, l'administrateur réseau qui a comme tâche d'analyser en détails les alertes générées se retrouve dans des situations difficiles.

Dans cette partie, notre objectif consiste à proposer des solutions à ces problèmes. Dans un premier temps, nous nous intéressons au problème de la détection d'intrusions où nous proposons d'utiliser les réseaux bayésiens qui sont des modèles graphiques dotés des algorithmes intéressants comme l'inférence, l'apprentissage automatique et classification. Une question fondamentale est la définition et l'extraction d'un nombre maximum d'informations (attributs) sur les attaques. C'est pourquoi, nous présentons l'outil de formatage que nous avons développé afin de transformer les données réseau brutes en données formatées et extraire des attributs que nous avons définis dans le but de décrire les activités normales ou anormales que présentent ces données.

Concernant la corrélation d'alertes, nos propositions consistent à développer une approche permettant de réduire le nombre d'alertes générées au sein d'un réseau en introduisant simplement les connaissances et les préférences de l'administrateur réseau. En fait, cette approche permet d'ordonner et classer les différentes alertes selon le niveau du gravité ou risque que chacune peut représenter tout en considérant les connaissances que l'administrateur a sur le fonctionnement de son système ainsi que ses préférences sur les alertes qu'il veut analyser ou ignorer. En effet, notre approche est basée sur l'utilisation d'une des logiques de la représentation des préférences que nous avons présentées dans le chapitre 3. L'idée générale consiste à présenter en premier uniquement les alertes qui satisfont les préférences et les connaissances de l'administrateur réseau. Par la suite, si nécessaire, les alertes les moins préférées seront présentées, et ainsi de suite.

Chapitre 5

Détection d'intrusions et corrélation d'alertes

Sommaire

5.1 Introduction

La sécurité informatique consiste à définir des politiques de sécurité pour assurer le bon fonctionnement des systèmes d'informations et développer des mécanismes afin de les protéger et prévenir les intrusions. Il existe par exemple, des pare-feux, qui permettent de contrôler l'accès des flux à l'aide de règles de filtrage, les mécanismes d'authentification qui permettent de prouver l'identité des utilisateurs (mots de passe, cartes à puces, etc.), les contrôles d'accès qui consistent à accorder aux utilisateurs uniquement des privilèges nécessaires pour accéder aux données et services ou des outils automatiques qui assurent l'analyse des vulnérabilités et des erreurs de configuration, etc.
Malheureusement, un système d'informations n'est pas sécurisé de façon sûre, car les politiques de sécurité définies et les mécanismes développés ont des limitations et ne permettent que réduire partiellement les risques de vulnérabilités et de failles.

Comme il n'est pas possible de garantir que le système d'informations ne contienne aucune faille ni vulnérabilité, les systèmes de détection d'intrusions (IDSs) sont utilisés pour détecter

des activités malveillantes au cas où les solutions préventives (pare-feux, contrôle d'accès, mécanismes d'authentification, etc.) sont déjouées. Les outils de détection d'intrusions proposés sont efficaces, mais leur utilisation individuelle ou coopérative présente plusieurs problèmes. Ce qui a motivé plusieurs recherches et travaux afin d'apporter d'autres solutions telle que la corrélation d'alertes qui est l'une des solutions prometteuses.

Dans ce chapitre, nous allons aborder deux points : la détection d'intrusions et la corrélation d'alertes. Dans la section 5.2, nous présentons la détection d'intrusions en général et les systèmes de détection d'intrusions. En particulier, nous donnons un exemple de système de détection d'intrusions dans la section 5.2.1. La section 5.2.2 présente les différentes techniques de détection utilisées. Par la suite, dans la section 5.3, nous abordons la corrélation d'alertes. Dans la section 5.4, nous passons en revue quelques approches de corrélation d'alertes. La section 5.5 conclut le chapitre.

5.2 Détection d'intrusions

La détection d'intrusions est un problème important qui préoccupe de nombreux chercheurs depuis les travaux fondateurs de ce domaine qui datent de 1980 [Anderson 1980]. L'objectif principal consiste à définir de nouveaux mécanismes et développer de nouvelles techniques permettant d'améliorer la sécurité des systèmes d'informations. Depuis, plusieurs systèmes et prototypes de détection sont proposés. Avant de présenter les types des IDSs existants et leur fonctionnement général, nous donnons d'abord la définition de certains concepts de la détection d'intrusions :

- **Vulnérabilité :** une vulnérabilité est une faiblesse ou un défaut dans la conception, l'implémentation et le fonctionnement d'un système informatique qui peut être exploité afin de violer la politique de sécurité. Par exemple, le protocole TCP présentait une faiblesse qui permet à un utilisateur mal intentionné d'envoyer des paquets de demande de fin de connexion.
- **Attaque :** une attaque consiste à exploiter une faille ou une vulnérabilité d'un système informatique dans l'objectif de violer la politique de la sécurité. Certaines attaques exploitent des failles du réseau, d'autres des failles de programmation, etc. Certaines d'autres visent à abuser d'actions autorisées (comme la majorité des déni de service). En général, plusieurs actions constituent une attaque comme la collecte d'informations, utilisation des informations récoltées pour accéder au sein d'un réseau, création d'un compte avec des droits de super utilisateur pour pouvoir observer ce qui est accessible et disponible sur le réseau dans le but de perturber le fonctionnement du système en détruisant par exemple des données.
- **Intrusion :** une intrusion peut être un événement ou une combinaison d'événements permettant de pénétrer un système sans autorisation, comme par exemple les tentatives des utilisateurs locaux d'accéder à de plus hauts privilèges que ceux qui leur sont attribués, ou tentatives des utilisateurs d'abuser de leurs privilèges.
- **Détection d'intrusions :** la détection d'intrusions consiste à analyser tout événement ayant lieu dans un système afin de détecter des activités malveillantes ou tentatives d'accès non autorisées. La détection d'intrusions peut être effectuée manuellement ou automatiquement. Lorsqu'il s'agit d'une détection manuelle, un administrateur procède à analyser les fichiers logs pour chercher des signes suspects pouvant indiquer des intrusions. Actuellement, ce type d'analyse n'est pas envisageable car le volume d'informations à analyser

est très important. La détection automatisée est effectuée par un système appelé *système de détection d'intrusions (IDS).*

- **Systèmes de Détection d'intrusions (IDSs) :** un IDS est un système qui comporte un ensemble de composants logiciels (parfois du matériel aussi) dont la fonction principale est de repérer des activités anormales ou suspectes qui tentent de compromettre l'intégrité (création, modification ou suppression illégale de l'information), la confidentialité (accès illégal à une information) et la disponibilité des ressources (empêchement des autres utilisateurs d'avoir un accès légal à certains services ou ressources) au sein d'un système d'informations. Par définition, un IDS n'a pas pour objectif d'empêcher que des attaques se produisent, mais de les détecter et entreprendre des actions comme par exemple les signaler à l'administrateur réseau afin qu'il prépare les contre-mesures les plus adaptées.
- **Règle de détection d'intrusions :** une règle de détection d'intrusions contient les informations nécessaires permettant de détecter une intrusion. Dans ces informations, se présentent les traces que les attaquants laissent lorsqu'ils attaquent le système.
- **Faux positif :** lorsqu'un IDS génère une alerte qui ne correspond pas à une attaque réelle, on l'appelle un *faux positif* ou *fausse alerte.*
- **Faux négatif :** un faux négatif est une intrusion réelle qui n'a pas été détectée par l'IDS.

Types des IDSs

Lors de la conception d'un système de détection d'intrusions, plusieurs paramètres sont pris en considération à savoir l'origine des données qui seront analysées, l'architecture du système (centralisée ou distribuée) et le mode d'analyse (temps réel ou en différé). Ainsi, il existe différents types d'IDSs :

1. Des IDSs réseaux (NIDS, Network-based IDS) : ce type d'IDSs permettent d'analyser les paquets circulant sur le réseau et détecter les intrusions en temps réel. Un NIDS écoute donc tout le trafic réseau grâce à une sonde qui permet de capturer le trafic réseau, puis l'analyser et génère des alertes si des paquets suspects sont identifiés.
2. Des IDSs systèmes (HIDS, Host-based IDS) : les IDSs systèmes, appelés aussi IDSs orientés hôte (poste de travail, serveur, etc.) analysent uniquement les données d'audit relatives aux activités du système d'exploitation et applications installées sur un hôte déterminé. Parmi ces données, nous trouvons par exemple des données *syslog* [Walters 1995] et données BSM [SunSoft 1995]. Ce type d'IDSs détecte un grand nombre d'attaques du fait que les données qu'ils analysent englobent l'activité entière du système.

Architecture et fonctionnement des IDSs

La figure 5.1 illustre les différents modules composant un système de détection d'intrusions. La première tâche d'un système de détection d'intrusions est d'abord de collecter des données provenant des activités des utilisateurs et de leurs interactions avec les services du système d'informations via des *capteurs. L'analyseur* est chargé d'analyser ces données afin de repérer des événements suspects. L'analyse est faite en adoptant l'une des méthodes de détection (nous abordons ce point dans la section 5.2.2). Dans le cas de la détection d'une activité suspecte, une alerte est transmise à un gestionnaire d'alertes (*manager*) qui se charge de la signaler à un administrateur réseau (*opérateur*). Notons que pour détecter les intrusions, les systèmes de détection d'intrusions utilisent trois catégories de sources de données : l'audit système, l'audit

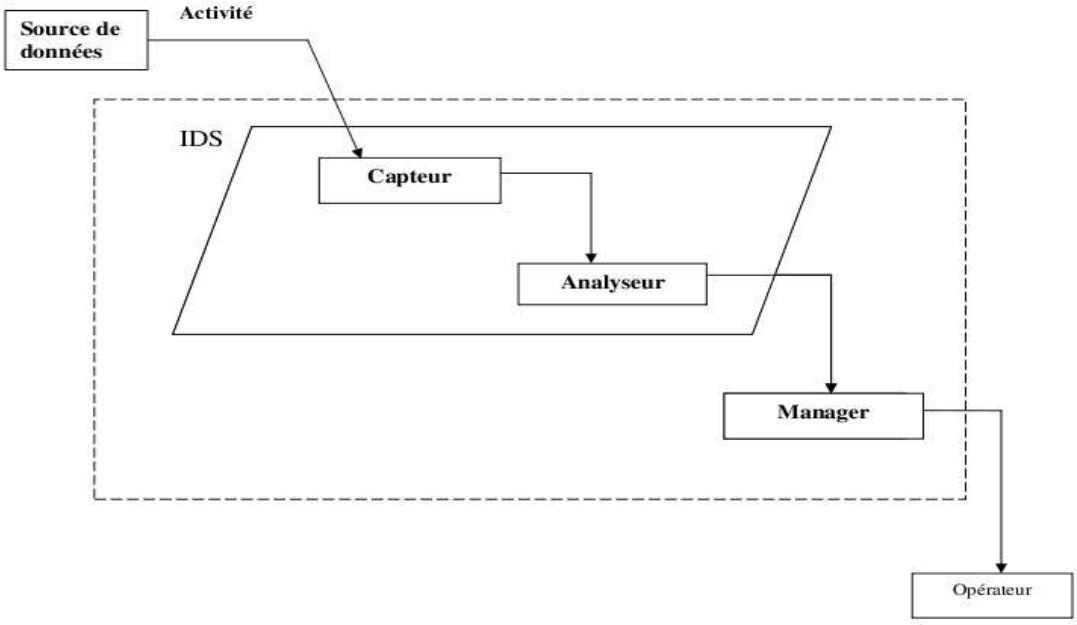

FIG. 5.1: Blocs fonctionnels d'un système de détection d'intrusions

applicatif et le trafic réseau. Le choix de la source de données dépend de l'activité que l'on veut surveiller et le type d'attaques qu'on veut détecter [Barse & Jonsson 2004], ce qui justifie le développement de plusieurs types d'IDSs.

5.2.1 Exemples de systèmes de détection d'intrusions

Nous présentons dans cette section un exemple de système de détection d'intrusions *Snort* [Koziol 2003].

Snort [Koziol 2003] est un système de détection d'intrusions réseau qui se base sur des signatures d'attaques connues. Il est capable d'analyser le trafic réseau en temps réel et de détecter de nombreux types d'attaques : buffer overflows, scans de ports, etc. Snort utilise une base de règles de détection (actuellement, plus de 1200 règles) spécifiée dans un fichier de configuration permettant de décrire les attaques que Snort va chercher dans les traces réseaux collectées. Les règles peuvent être classées selon le type de trafic. Ainsi, nous trouvons des règles spécifiques aux services Web, aux appels de procédures distantes (RPC), etc.
Une caractéristique importante de Snort concerne la possibilité de formuler ses propres règles pour pouvoir analyser un type de trafic particulier et d'adapter au mieux Snort au réseau qu'il doit surveiller et protéger. Une règle Snort se compose de deux parties distinctes : *l'entête et les Options.*

La partie *entête* permet de spécifier deux paramètres :

- *Le type de la règle de détection :* Snort définit cinq types de base : *alert, log, pass, active et dynamic.* Le type *alert*, par exemple, enregistre le paquet qui répond aux critères indiqués dans la règle et génère une alerte pour avertir l'administrateur réseau.
- *Les champs de base nécessaires au filtrage :* ces champs concernent le protocole (tcp, udp, icmp et ip), les adresses IP source et destination du paquet et les numéros de port utilisés

par la source et la destination (des noms de service peuvent être également utilisés : ftp, http, etc.). Des opérateurs de direction (->, <>) sont aussi spécifiés dans le but d'indiquer le sens du paquet sur lequel sera appliquée la règle de détection.

La partie *Options* peut contenir plusieurs options sauvegardées dans une structure appelée *OTN* (Option Tree Node). Chaque option spécifie un critère de détection. Parmi les options, citons par exemple :

- msg : spécifie le message qui sera associé à l'alerte,
- priority : définit la sévérité de l'alerte (1, 2, 3, etc.).
- content : représente la chaîne de caractères qui doit être présente dans le paquet pour déclencher l'action décrite dans la règle.
- ttl (durée de vie) : spécifie la valeur TTL[3] du paquet.

Ci-après un exemple de règle Snort :

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg :"FTP .rhosts") ;
    flow :to_server,established ; content :".rhosts" ; metadata :service ftp ;
reference :archnids,328 ; classtype :suspecious-filename-detect ; sid :335,rev :6 ;)
```

FIG. 5.2: Exemple d'une règle Snort (règle ftp)

Cette règle permet de générer un message d'alerte "FTP .rhosts" lorsque le trafic à destination des serveurs FTP (définis dans le fichier de configuration de Snort, *snort.conf*)), contient le mot ".rhosts" (spécifiée par l'utilisation du mot-clé *content*). Cet événement indique un accès suspect au fichier système ".rhost" (classe suspecious-filename-detect).

Cinq modules interviennent pour le fonctionnement de Snort : le module *Renifleur* utilisé pour capter les paquets réseaux, *Décodeur de paquets* qui permet de structurer le contenu des paquets, *Pré-processeur* qui s'occupe du prétraitement du trafic, *Module de détection* qui analyse le trafic en appliquant les règles de détection et *Module de sortie* qui permet par exemple de mettre les alertes générées dans une base de données. Snort se base sur l'analyse syntaxique des règles que l'administrateur prend en considération dans le fichier de configuration. Si un paquet suspect est détecté dans le trafic analysé, alors Snort va le signaler à l'administrateur.

5.2.2 Méthodes de détection

Deux approches sont principalement utilisées par les systèmes de détection d'intrusions. La première est appelée *approche comportementale*. Elle consiste à vérifier la conformité des activités des utilisateurs par rapport à un profil normal prédéfini. La seconde approche est appelée *approche par signatures*, consistant à rechercher les signatures d'attaques connues dans les données analysées.

[3]Time To Live

Approche comportementale

L'approche comportementale (*Anomaly detection, en anglais*) est considérée comme la première technique à être utilisée en détection d'intrusions [Anderson 1980]. Elle se base sur la définition d'un profil qui représente les différentes activités normales du système dans lequel cette technique a été mise en œuvre. Elle peut être appliquée à des utilisateurs, applications et services réseaux, etc. Toute déviation par rapport au profil de comportement normal défini sera interprétée comme une éventuelle intrusion. Le profil défini sera utilisé comme une référence. Ensuite, lors de la phase de surveillance par un système de détection d'intrusions, un autre profil représentant le comportement réel des utilisateurs ou applications est défini en temps réel et comparé au profil défini auparavant. La comparaison se fait généralement en utilisant des fréquences, moyennes, écart-types, etc. pour évaluer l'écart entre le nouveau profil calculé et le profil de référence. En général, la détection d'anomalies dans l'approche comportementale fonctionne en deux phases : une phase d'apprentissage et une phase de détection. Au cours de la phase d'apprentissage, le système apprend à reconnaître le comportement normal, et au cours de la phase de détection, il identifie les comportements anormaux. L'approche comportementale offre la capacité de détecter de nouvelles attaques lorsque les attaques provoquent des déviations par rapport aux comportements normaux de référence. Cependant, son principal inconvénient concerne le nombre important de fausses alertes générées dû aux changements dans les comportements des utilisateurs, des services, etc.
Cette approche peut faire appel à plusieurs méthodes pour la détection. Nous citons par exemple :

- **Méthodes statistiques :** ces méthodes sont utilisées dans le but de quantifier les paramètres liés aux utilisateurs du système ou applications, comme le taux d'occupation de la mémoire, l'utilisation des processeurs, la durée et l'heure des connexions, les sites les plus visités, etc. Ainsi, le profil statistique défini ou calculé sera utilisé comme profil de référence pour la détection. IDES [Denning 1987] et EMERALD [Porras & Neumann 1997] sont des exemples de systèmes de détection d'intrusions qui utilisent des méthodes statistiques.

- **Méthodes basées sur des règles définies (systèmes experts) :** ces méthodes utilisent un ensemble de règles pour représenter les comportements normaux. Le moteur d'inférence utilise la base de règles pour identifier les anomalies qui existent dans la base de faits. ComputerWatch [Dowell & Ramstedt 1990] est un exemple d'outil qui utilise un système expert dans le cadre de l'approche comportementale.

- **Méthodes basées sur des techniques de fouilles de données :** les techniques de fouilles de données permettent de construire un profil normal à partir de l'historique des données d'audit représentant des activités normales. Des attaques sont identifiées lorsque des déviations par rapport au profil normal établi sont observées. Comme exemples d'IDSs de cette catégorie, on cite : ADAM [Barbará *et al.* 2001] et eBayes [Valdes & Skinner 2000].

- **Analyse de protocoles :** cette méthode vérifie si le comportement d'un trafic circulant sur le réseau est conforme aux spécifications des protocoles (chaque protocole a un "RFC"[4] qui le spécifie). Les systèmes qui utilisent cette méthode considèrent par exemple anormal tout trafic présentant des paquets contenant de fausses combinaisons

[4]Request For Comments

de flags TCP, des adresses non autorisés, des numéros de ports interdits, etc.

Approche par signatures

L'approche par signatures (*misuse detection, en anglais*) s'appuie sur les différentes techniques utilisées par les attaquants pour déduire des scénarios typiques. Elle utilise des signatures d'attaques (une chaîne de caractères, une taille de paquet inhabituelle, accès à un fichier système, etc.). Cette technique nécessite une connaissance a priori sur les attaques à détecter : une alarme est émise lorsqu'une trace ou signature d'une attaque connue est détectée. Le principal inconvénient de cette approche est que seules les attaques connues dans la base de signatures seront détectées. Il est donc nécessaire de mettre régulièrement à jour la base de signatures. Par contre, elle est satisfaisante du point de vue du taux de détection dans les attaques connues qui est généralement assez élevé. De plus, le nombre de fausses alertes qu'elle génère n'est pas important par rapport l'approche comportementale. L'approche par signatures utilise des techniques différentes, dont les principales sont :

- **Recherche de motifs (pattern matching) :** la méthode la plus connue est la recherche de motifs et séquences particulières au sein des flux de données. L'IDS comporte alors une base de signatures où chacune contient des informations concernant une attaque (protocole et ports utilisés, le motif qui permettra de reconnaître les paquets suspects, etc.). La figure suivante montre un exemple d'une signature Snort :

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
        (msg :"WEB-MISC apache directory disclosure attempt" ;
            flow :to_server,established ; content :"////////" ;
    reference :bugtraq,2503 ; classtype :attempted-dos ; sid :1156, rev :9 ;)
```

FIG. 5.3: Exemple d'une signature Snort

  Cette signature permet de générer un message d'alerte "WEB-MISC apache directory disclosure attempt" lorsque le trafic à destination des serveurs HTTP définis dans le fichier de configuration de Snort (snort.conf) contient la chaîne "////////" (spécifiée par l'utilisation du mot-clé *content*). Cet événement suspect sera classé dans la classe attempted-dos. Snort [Koziol 2003] et Bro [Paxson 1999] sont des exemples d'IDSs utilisant cette méthode de détection.

- **Systèmes experts :** un système expert comprend une base de connaissances contenant des règles concernant non seulement des signatures d'attaques connues, mais aussi les vulnérabilités connues du système. La base de connaissances est construite en se focalisant sur l'expérience de l'administrateur réseau. Une activité suspecte déclenche ces règles et engendre l'émission d'une alerte. ASAX [Habra *et al.* 1992] et P-BEST [Lindqvist 1999] sont parmi les IDSs les plus connus utilisant ce mécanisme.

- **Fouille de données :** l'utilisation de l'approche fouille de données pour la détection d'intrusions a été largement explorée dans de nombreux travaux étant donné qu'il existe des algorithmes de classification et apprentissage automatique (comme les réseaux bayésiens, les réseaux de neurones et les arbres de décisions) pouvant à partir d'un jeu de données

représentatif des activités normales ou anormales apprendre, sans connaissances a priori, les signatures d'intrusions et d'appliquer le modèle ainsi élaboré aux activités courantes. JAM [Lee *et al.* 1998], MADAM ID [Lee *et al.* 2000] sont deux exemples d'IDSs dont l'approche par signatures adoptée se base sur des techniques de fouille de données.

D'autres techniques ont été également proposées telles que l'utilisation des réseaux de Petri dans les systèmes IDIOT [Crosbie *et al.* 1996], les algorithmes génétiques [Mé 1994] et techniques d'analyse à état-transistion dans STAT [Porras 1993] et NetStat [Vigna & Kemmerer 1999].

Limites et problèmes

Les outils de détection d'intrusions ainsi que les approches adoptées sont nécessaires pour assurer la sécurité des systèmes d'informations. Cependant, ils ont chacun des limites et leur utilisation pose plusieurs problèmes. Lors de l'utilisation de l'approche comportementale, l'apprentissage peut être inexact dans le cas où le profil défini, supposé normal, peut contenir des attaques inconnues. Il devient alors difficile de juger la nature réelle des activités au sein du système. Le second problème qui limite sérieusement les systèmes de détection d'intrusions adoptant l'approche comportementale est le taux de fausses alertes trop élevé dû au principe de détection [Lee & Xiang 2001]. En effet, cette approche considère tout ce qui est anormal comme une intrusion, alors qu'une anomalie par rapport au profil normal peut être une activité totalement légale et normale mais inconnue pour l'IDS. De plus, un attaquant peut modifier graduellement le profil normal afin d'obtenir un profil qui lui permettra de lancer ses attaques sans qu'elles ne soient détectées. Pour les systèmes de détection d'intrusions qui se basent sur l'approche par signatures, le principal inconvénient est que seules les attaques connues dans la base de signatures seront détectées, ce qui nécessite une mise à jour fréquente de la base de signatures. Des connaissances expertes sont également nécessaires pour pouvoir écrire des signatures d'attaques efficaces. Un autre inconvénient de l'approche par signatures est que la moindre différence dans l'attaque par rapport à la signature connue, le système ne pourra pas détecter cette attaque.

5.3 Corrélation d'alertes

Les deux approches de détection présentées dans les sections précédentes sont indispensables pour la surveillance et la protection d'un système d'informations. En revanche, leur utilisation pose plusieurs problèmes. Ainsi, quelle que soit l'approche de détection utilisée par les IDSs, l'administrateur réseau est souvent confronté à de nombreux problèmes. Le problème majeur réside dans les grands volumes d'alertes que les IDSs produisent. L'administrateur réseau qui a comme tâche d'analyser et de prendre des décisions appropriées se retrouve rapidement débordé. Ainsi, l'analyse des alertes est certainement moins efficace parce que l'administrateur a besoin de beaucoup de temps et laisse certainement passer des attaques. Plusieurs approches de corrélation d'alertes ont été proposées dans le but de remédier à ce problème.

La corrélation d'alertes est une branche de la détection d'intrusions qui est apparue récemment comme une technique complémentaire aux différentes techniques de détection utilisées. Elle est considérée comme un mécanisme indispensable vu les grands volumes d'alertes générées quotidiennement. Les approches de corrélation d'alertes développées ont exploité plusieurs solutions à savoir des méthodes probabilistes, logiques ou autres, dont le but principal consiste à découvrir les relations ou similarités qui peuvent exister entre les différentes alertes afin de regrouper et supprimer les alertes redondantes. D'une manière générale, *la corrélation*

d'alertes est définie comme l'interprétation conceptuelle de plusieurs alertes afin, d'une part, de leur attribuer une meilleure sémantique et, d'autre part, de réduire le volume important d'alertes [Jakobson & Weissman 1993].
Avant de présenter quelques objectifs principaux de la corrélation d'alertes, nous décrivons d'abord la structure des alertes produites par les outils de détection d'intrusions.

5.3.1 Format des alertes

La difficulté de détecter de nouvelles attaques a amené les administrateurs à multiplier les outils de détection et de surveillance. Or, la multiplication des outils de détection va générer encore des quantités énormes d'alertes issues des systèmes hétérogènes. Afin d'offrir une vue globale de l'activité des outils de surveillance, un format standard des alertes devient nécessaire afin d'avoir un minimum de cohérence au niveau des informations contenues dans les alertes. Ainsi, plusieurs formats sont proposés : IDMEF (Intrusion detection Message Exchange Format) [Curry & Debar 2001] est l'exemple de format d'échange d'alertes le plus utilisé. Il a été développé par le groupe IDWG (Intrusion Detection Working Group). IDMEF divise les informations contenues dans les alertes en plusieurs catégories :

- *Événement (attaque) :* le champ événement présente le nom de l'action effectuée par l'attaquant dans le système d'informations surveillé. Si nous prenons l'exemple d'une alerte générée par un analyseur qui se base sur l'approche par signature tel que Snort, l'événement que l'alerte représente est associé à un identifiant unique (appelé identifiant de signature (SID)) et d'un message textuel. Notons que le format des identifiants associés aux événements que les alertes décrivent dépend du type de l'analyseur qui les produit, ce qui pose un problème majeur pour la corrélation des alertes. Il est donc nécessaire d'utiliser un dictionnaire permettant d'établir des correspondances.
- *Attaquant (source) :* ce champ représente des informations relatives à l'origine de l'événement qui a provoqué la génération de l'alerte au sein du système d'informations. Ces informations servent à identifier les attaquants et peuvent concerner : les identifiants d'utilisateurs, les nœuds réseaux, les identifiants des processus et les services réseaux.
- *Victime (destination) :* les victimes sont des entités du système d'informations visées par les attaques. Il existe cinq classes permettant d'identifier les victimes : les identifiants d'utilisateurs, les nœuds réseau, les identifiants des processus , les services réseaux et les fichiers concernés par l'attaque.
- *Date :* ce champ représente des informations temporelles concernant les alertes. Comme défini dans le format IDMEF, il existe trois types d'informations temporelles :
  - *create time* : précise la date à laquelle l'alerte a été créée par l'analyseur,
  - *detect time* : précise la date de réception du premier événement qui compose l'attaque,
  - *analyzer time* : concerne la date courante de l'analyseur ayant généré l'alerte. Elle est utilisée pour ajuster les horloges des analyseurs.

Les alertes peuvent contenir également d'autres informations comme le niveau de sévérité d'alerte qui est un entier positif indiquant le niveau de gravité de l'attaque.

5.3.2 Objectifs de la corrélation d'alertes

Parmi les raisons fondamentales qui ont amené les chercheurs à définir des mécanismes de corrélation d'alertes, nous pouvons citer :

Réduction du volume d'alertes

L'objectif fondamental des techniques de corrélation d'alertes consiste à réduire la quantité des alertes que l'administrateur réseau doit analyser et traiter. En effet, les mécanismes de détection qui sont utilisés par les IDSs telles que l'approche comportementale et l'approche par scénarios restent insatisfaisants dans le sens où non seulement ils n'arrivent pas toujours à détecter par exemple des attaques dont le comportement diffère légèrement de celui connu dans la base d'apprentissage dans le cas d'une approche comportementale, ou aussi des attaques dont la signature n'est pas enregistrée dans la base de signatures dans le cas où l'approche par scénarios est utilisée, mais ils génèrent également des quantités d'alertes ingérables par l'administrateur réseau.
La réduction du nombre d'alertes à analyser s'avère indispensable dans la mesure où une grande partie de ces alertes sont de fausses alertes (si on prend l'exemple du trafic normal de DARPA'99 [Darpa 1999], l'IDS Snort génère plus de 60000 alertes). On trouve également beaucoup d'alertes répétitives ; cela est dû par exemple à l'observation d'une chaîne de caractère suspecte au niveau de chaque paquet de la même connexion. Ainsi, si une connexion anormale contient par exemple 100 paquets, 100 alertes peuvent être générées, alors que réellement, une seule alerte (avec des informations nécessaires telles que les adresses IP source et destination) peut être suffisante pour signaler la présence d'une activité suspecte. Il existe aussi beaucoup d'alertes réellement identiques car elles représentent le même événement, mais vu qu'elles sont générées par plusieurs IDSs, elles ne sont pas considérées identiques, où par exemple le nom associé à l'événement détecté peut varier pour chaque IDS, la quantité d'informations que contient chaque alerte peut aussi varier.
La réduction du volume important d'alertes peut se faire de différentes manières à savoir la suppression des fausses alertes, le regroupement des alertes qui se réfèrent à une même activité, l'extraction des liens logiques ou autres qui peuvent exister entre les alertes, etc. Notons que dans la plupart des travaux existants, lorsque le problème de la corrélation d'alertes est abordé, les concepts "duplication, regroupement, suppression, déduction, fusion et agrégation" sont aussi utilisés.

Amélioration de la qualité du diagnostic

Il s'agit d'améliorer le niveau de précision en quantité et qualité des informations que contiennent les alertes, car lorsque les systèmes de détection d'intrusions génèrent des alertes, elles sont souvent de très bas niveau, et parfois contiennent des informations insuffisantes et imprécises au sujet de l'activité anormale qu'elles décrivent. Il existe des informations ou critères intéressants pour évaluer les risques que représentent les attaques, mais ils sont souvent manquants. Ces informations peuvent concerner par exemple une vulnérabilité présente réellement dans le système d'informations, identification d'outils d'attaque, etc. Par exemple, si un utilisateur tente d'accéder la première fois avec un login incorrect, alors une alerte "login incorrect" est levée sans plus de précision, alors que cet utilisateur peut être interne au système et ce n'est qu'un oubli. Donc, si nous pouvons rajouter une information qui indique par exemple, le nombre de fois qu'un utilisateur a tenté d'accéder avec un login incorrect, cela peut être intéressant.

Suivi des plans des attaques complexes

Une attaque se caractérise généralement par un *plan d'attaque*, où plusieurs actions sont effectuées dans un ordre bien défini. Une attaque est donc constituée de plusieurs sous-attaques. Certaines sous-attaques cherchent à obtenir des informations sur le système d'informations en

exécutant les premières actions pour le modifier ou fournir ces informations à d'autres sous-attaques pour accomplir le reste des actions dans le but d'atteindre un certain état cible. Ainsi, chaque sous-attaque ou chaque action suspecte qui s'exécute peut donner lieu à plusieurs alertes. Pour pouvoir reconnaître les différentes actions d'une attaque, plusieurs méthodes ont été mises au point, nous citons par exemple [Cuppens & Miège 2002, Templeton & Levitt 2000, Ning *et al.* 2002, Cuppens & Ortalo 2000].

5.4 Approches de corrélation d'alertes

Nous distinguons trois principales catégories d'approches de corrélation d'alertes : les approches qui se basent sur les mesures de similarité entre les attributs des alertes
[Valdes & Skinner 2001, Staniford *et al.* 2002, Julisch 2003], les approches qui se basent sur les scénarios d'attaques connues [Eckmann *et al.* 2002, Morin *et al.* 2002] et les approches qui se basent sur des conditions préalables et les conséquences des relations [Cuppens & Miège 2002, Ning *et al.* 2002]. Chacune de ces approches définit un mécanisme bien particulier pour remédier aux différents problèmes rencontrés. Nous détaillons ces trois approches dans ce qui suit.

5.4.1 Approches de corrélation à base de similarité entre attributs

Le principe de base de ce type d'approches consiste à traiter les alertes dans leur globalité afin d'exploiter les similarités qui peuvent exister entre les attributs qui les identifient. Ces attributs peuvent concerner, par exemple, certains champs d'une alerte tels que ceux qui sont définis dans le format IDMEF [Curry & Debar 2001]. Ces approches sont classées comme des techniques de corrélation implicites. Nous présentons, en particulier, l'approche probabiliste de Valdes et Skinner [Valdes & Skinner 2001].

Corrélation probabiliste d'alertes

Dans [Valdes & Skinner 2001], les auteurs ont proposé une méthode probabiliste qui consiste à définir une fonction pour calculer des mesures de similarités entre deux alertes. Cette fonction prend comme paramètres deux alertes (une nouvelle alerte et un groupe d'alertes représenté par une alerte, appelée meta-alerte) et retourne un nombre réel dans l'intervalle [0, 1] qui correspond à la valeur de similarité des deux alertes étant donné un attribut. Les attributs qui représentent les alertes sont : la catégorie de l'attaque (DoS (Denial Of Service) et integrity violation par exemple), les adresses IP source et destination, l'identifiant de l'IDS ou l'analyseur ayant généré l'alerte et la date d'occurrence.
Étant donnée une alerte candidate X et une méta-alerte Y, la valeur de similarité entre ces deux alertes étant donné un attribut i est définie par :

$$SIM(X,Y) = \frac{\sum_i E_i SIM(X_i, Y_i)}{\sum_i E_i} \tag{5.1}$$

E_i correspond au degré de similarité attendue concernant l'attribut i (expectation of similarity). X_i est la valeur de l'attribut i de l'alerte X. Y_i est une liste de valeurs que prend l'attribut i dans les alertes participant à la méta-alerte Y. SIM est la matrice de similarité utilisée dans le calcul de similarité pour l'attribut spécifiant la classe de l'événement associée à l'alerte. Dans le cas où la valeur de similarité est en dessous de la valeur attendue(E_i), alors la nouvelle alerte ne peut pas être candidate à la corrélation. Dans ce cas, l'alerte forme une nouvelle méta-alerte.

Notons que des caractéristiques propres à chaque attribut sont prises en compte lors du calcul de la mesure de similarité. Par exemple, pour calculer la similarité entre deux adresses IP, il est préférable qu'elles appartiennent au même sous-réseau, c'est-à-dire que deux adresses IP seront similaires si elles appartiennent au même sous-réseau.
Le principal inconvénient de cette méthode concerne la matrice de similarité qui est définie en fonction de certains attributs des alertes et non pas en fonction des classes d'événements associées aux alertes, alors que nous avons des connaissances a priori sur les classes des événements et que l'importance de certains attributs d'une alerte est fonction de la classe d'événements qui lui est associée. En outre, cette méthode ne concerne que certaines informations par rapports à celles de IDMEF.

Il existe plusieurs autres approches de ce type. Dans [Cuppens & Autrel 2005], les auteurs ont présenté une méthode pour l'agrégation d'alertes basée également sur une fonction de similarité. Dans [Dain & Cunningham 2001], les auteurs ont proposé une méthode similaire à celle de Valdes et Skinner, mais ils ont considéré uniquement trois attributs : la catégorie d'attaque, l'adresse IP source et la date d'occurrence. La similarité entre alertes est le produit des similarités entre attributs. Dans [Cuppens 2001], Cuppens propose une technique qui utilise des prédicats logiques pour définir la similarité entre les alertes. Des règles expertes doivent être définies afin de comparer les attributs des alertes. Par exemple, pour comparer deux événements générés par deux IDSs, une bibliothèque de correspondance entre les noms des événements a été définie. D'autres approches appartenant à la même catégorie sont proposées dans [Julisch 2003, Staniford *et al.* 2002, Manganaris *et al.* 2000], etc.

Ces approches sont classées dans une catégorie de corrélation implicite basée généralement sur des outils de fouilles de données. Les alertes sont traitées dans leur totalité et les relations de corrélation entre les événements que décrivent les alertes sont généralement extraites à partir d'un ensemble de données. Ces approches sont efficaces puisqu'elles permettent de découvrir les liens entre les alertes générées par plusieurs IDSs. Cependant, elles ne peuvent pas découvrir toutes les relations possibles.

5.4.2 Approches basées sur les scénarios d'attaques

Cette catégorie d'approches se base sur les scénarios d'attaques connues (c-à-d. les différentes étapes ou actions qu'un attaquant utilise pour réaliser ses attaques). Les scénarios d'attaques doivent être définis pour les comparer par la suite aux alertes générées. Dans la plupart des travaux appartenant à cette catégorie d'approches, un scénario d'attaque peut être spécifié soit par un langage de description d'attaques soit appris à partir de données d'apprentissage par l'utilisation d'approches de fouilles de données. Nous présentons d'abord les langages de description des attaques.

Langages de description d'attaques

Pour détecter tout événement suspect, les systèmes de détection d'intrusions doivent coopérer afin de pouvoir reconnaître les différentes actions exécutées et faisant partie de la stratégie de l'attaque. Les informations qu'une alerte représente dépendent de la technique de détection utilisée dans le sens où le contenu des alertes générées par deux IDSs différents comme Bro et Snort n'est pas totalement identique. Il est souvent difficile de retrouver ou reconnaître la stratégie suivie par une attaque du fait que, d'une part, les informations contenues dans chaque alerte

ne donnent pas des renseignements précis sur la sémantique des événements et, d'autre part, les liens entre les événements ou les actions sont quasiment absents dans le contenu des alertes. Afin de pouvoir décrire le comportement d'une attaque, il est nécessaire de faire appel à ce que l'on appelle *langage de description d'attaques*.
Un langage de description d'attaques permet de modéliser la stratégie qu'un attaquant applique pour réaliser ses attaques. Il permet également d'analyser les relations entre les différentes attaques dans le but d'identifier les attaques coordonnées contre le système. Dans [Vigna *et al.* 2000], les auteurs ont classifié les langages de description d'attaques en six catégories :

- *Langages de description d'événements :* ces langages permettent de décrire le format des événements utilisés dans le processus d'analyse et de détection. Il existe plusieurs langages, mais chacun propose un format différent. Voir [Bishop 1995, Jacobson *et al.* 2000] pour plus de détails.
- *Langages de réponse :* les langages de ce type sont utilisés pour spécifier des mécanismes ou des contre-mesures permettant de réagir lorsqu'un IDS détecte l'attaque.
- *Langages de synthèse :* ces langages décrivent le format des alertes générées par les IDSs. Ces langages peuvent être considérés aussi comme des langages d'événements à un niveau plus élevé. A titre d'exemple, ils peuvent être utilisés comme des entrées pour les outils de corrélation. CISL (Common Intrusion Specification Language) [CISL 2000] et IDMEF (Intrusion Detection Message Exchange Format) [Curry & Debar 2001] sont des exemples de ce type de langages.
- *Les langages de corrélation :* ces langages prennent en considération les alertes générées par les différents IDSs afin d'identifier les attaques coordonnées et complexes lorsque des liens de corrélations existent.
- *Langages de description d'exploits :* ces langages permettent de décrire toutes les étapes nécessaires pour qu'une attaque se réalise afin de pouvoir la rejouer sur une machine. Dans [Cuppens & Ortalo 2000], [Deraison 2000], les auteurs ont proposé des langages de ce type.
- *Langages de détection :* ces langages fournissent des mécanismes pour identifier les actions qui se manifestent lors de l'exécution d'une attaque au sein d'un système. Comme exemples de ce type de langages, nous citons les langages de Bro [Paxson 1999], Snort [Koziol 2003], STATL [Eckmann *et al.* 2002], GASSATA [Mé 1998], etc. En particulier, les langages de Snort et Bro sont considérés comme des langages d'expression de signatures d'attaques.

Dans ce qui suit, nous donnons une brève présentation du langage de description d'attaques AdeLe.

AdeLe (Attack Description Language)

AdeLe [Michel & Mé 2001] est un langage de description d'attaques proposé dans le cadre du projet MIRADOR. Ce langage permet de définir une base de données d'attaques connues en spécifiant leurs propres scénarios. L'objectif principal de *AdeLe* consiste à combiner toutes les connaissances disponibles sur l'attaque en une seule connaissance afin de représenter une description de haut niveau de l'attaque. La description d'attaques donnée par AdeLe est composée de trois étapes :

- **La partie exploitation (EXPLOIT) :** cette partie représente en détail l'attaque du point de vue des conditions nécessaires pour son exécution, les différentes étapes, le code ainsi que le résultat de son exécution, etc. Cette première partie est donc composée de trois parties : *pré-conditions*, *code* et *post-conditions*. Dans la partie *pré-conditions*, sont exprimées les conditions nécessaires à la réalisation de l'attaque. Il s'agit la plupart du temps, des connaissances sur les vulnérabilités qui peuvent être présentes au niveau de la cible visée par l'attaque. La partie *code* permet de spécifier le langage utilisé pour exprimer le code source de l'attaque et sélectionner l'interpréteur adéquat à son exécution. Le langage spécifié peut être un des langages de programmation habituels comme C ou C++. Une description informelle de l'attaque à base de texte peut être également utilisée.
  Notons qu'un langage pour la description de la réalisation d'attaques a été spécialement développé dans [Michel & Mé 2001]. Ce langage est appelé *EDL (Exploit Description Language)*. Concernant la partie *post-conditions*, des effets ou des résultats de l'exécution de l'attaque sont exprimés. Ces effets peuvent concerner l'obtention de privilèges réservés à l'administrateur par exemple, modification de fichiers, réalisation d'un déni de service, etc.

- **La partie détection (DETECTION) :** dans cette partie, les auteurs ont proposé un langage de haut niveau qui permet, à la fois, d'écrire des signatures permettant de détecter des attaques élémentaires de bas niveau, et d'exprimer des scénarios complexes qui correspondent à des attaques connues. Cette partie se compose de trois sous parties : *DETECT*, *CONFIRM* et *REPORT*.
  Dans la partie *DETECT*, plusieurs éléments doivent être spécifiés : des noms pour les alertes (ou événements) qui vont se produire lors de l'exécution de l'attaque, des relations temporelles ou l'enchaînement entre les événements produits dans le but de définir le scénario de l'attaque et des caractéristiques de chaque événement ainsi que des contraintes contextuelles entre les événements qui appartiennent à la même attaque. Pour décrire un scénario global d'attaque (c-à-d. enchaînement d'alertes ou d'événements), les auteurs ont utilisé une combinaison de 8 opérateurs :

  - *l'opérateur ";" :* permet d'exprimer une séquence d'événements.
  - *Non_ordered<events> :* lorsqu'un ensemble d'événements appartenant à une attaque sont produits dans un ordre aléatoire, cet opérateur peut être utilisé. Il définit un intervalle temporel borné par les occurrences du premier et du dernier événements.
  - *One_among<event> :* exprime qu'un événement parmi un ensemble d'événements doit être utilisé dans l'une des étapes de l'attaque.
  - *Subset_of<event> :* exprime que un ou plusieurs événements d'un ensemble d'événements doivent être observés.
  - *"ˆ" :* cet opérateur exprime l'apparition d'un même événement plusieurs fois (c-à-d. répétition). $E\hat{}n$ indique que l'événement E doit être observé n fois.
  - *" := " :* permet de nommer une occurrence particulière d'un événement. Par exemple, pour isoler la 7^{eme} occurrence de 10 événements similaires, nous pouvons écrire :
    $E\hat{}10$; EVT1 :=E ; $E\hat{}3$
  - *WITHOUT :* cet opérateur exprime qu'un événement particulier ne doit pas être observé dans un intervalle d'événements spécifié. Par exemple, E0 ; E1 ; E2 WITHOUT F signifie que l'événement F ne doit pas être observé entre E0 et E2.
  - *Time constraints (Contraintes temporelles) :* cet opérateur exprime des contraintes temporelles entre événements. Par exemple : (E2.Time.time - E1.Time.time) <= "2mn"

La partie *CONFIRM* exprime ce qu'il faut vérifier sur le système surveillé lorsque l'attaque est exécutée avec succès. Pour cela, un ensemble de fonctions booléennes sont fournies. Par exemple, la fonction Unreachable_Machine(<IP adress>) retourne la valeur vraie lorsque la machine spécifiée ne répond pas. La partie REPORT représente le format de l'alerte générée lors de la détection d'une attaque décrite en ADeLe.

- **La partie réponse (RESPONSE) :** cette partie exprime les contres-mesures qui sont prises pour réagir lors de la détection de l'attaque. Plusieurs fonctions sont proposées telle que la fonction
  Close_Port(*target_ip*,**"TCP"|"UDP"**,*port_number*) qui permet de fermer le port utilisé par l'attaquant jusqu'à ce que la vulnérabilité soit corrigée.

Dans ce qui suit, nous présentons deux approches de corrélation d'alertes appartenant à la deuxième catégorie.

Algorithme d'agrégation et de corrélation (AC)

Dans [Debar & Wespi 2001], les auteurs ont intégré un algorithme d'agrégation et de corrélation (AC) dans une interface implémentée sous la console Tivoli (TEC pour Tivoli Entreprise Console). Le modèle est organisé en quatre différentes couches :

- La couche sonde : contient des informations sur l'ensemble des alertes générées, comme le nom de la sonde qui les génèrent.
- La couche cible : contient des informations sur la (les) cible(s) d'une alerte.
- La couche source : contient des informations sur la source d'une alerte.
- La couche cible détaillée.

L'objectif de cet algorithme de corrélation consiste à regrouper les alertes selon le type des relations qu'elles décrivent entre elles. Nous distinguons deux techniques de regroupement :

- *relations de corrélation*
- *relations d'agrégation.*

Dans la première technique (*relations de corrélation*), la corrélation est réalisée par le regroupement des alertes relatives à la détection d'un même événement (alertes qui sont générées par plusieurs IDSs mais décrivant la même attaque par exemple) à base d'un ensemble de liens qui existent entre les alertes. Dans ce cas, l'algorithme AC utilise des règles explicites programmées a priori ou extraites des fichiers de configuration pour définir des *alertes dupliquées* et des *conséquences.* Les alertes dupliquées et les conséquences doivent être spécifiées dans le fichier de configuration où chacune est associée d'un ensemble de termes. Par exemple, une alerte dupliquée est définie par quatre termes qui sont :

- **Classe d'alerte initiale :** indique la classe d'alerte qui est déjà reçue.
- **Classe d'alerte dupliquée :** correspond à la classe d'alerte candidate, c'est-à-dire la classe de la nouvelle alerte reçue qui sera évaluée dans le but de la dupliquer avec la première.
- **Liste d'attributs :** ce terme concerne la liste des attributs associés aux alertes devant être égaux pour dupliquer ou associer les alertes.
- **Niveau de sévérité :** correspond au niveau de gravité d'alerte dupliquée que l'algorithme AC ajoute dans le fichier de configuration pour l'utiliser lorsqu'une nouvelle alerte est reçue.

Lorsqu'une nouvelle alerte est reçue, l'algorithme AC cherche dans le fichier de configuration des alertes dupliquées dont la classe d'alerte (deuxième terme) est similaire à la classe de la nouvelle alerte. Il cherche par la suite les alertes déjà reçues où la classe d'alerte correspond à celle de l'alerte reçue en premier. Si l'alerte initiale est trouvée, alors elle est associée ou regroupée avec la nouvelle reçue et le niveau de gravité est associé avec l'alerte dupliquée pour l'utiliser plus tard dans le processus de regroupement.

Notons que la définition d'alertes dupliquées implique un certain ordre dans la réception des alertes. Cette définition permet de donner des renseignements sur les alertes que l'on peut recevoir lors de la détection d'un événement ainsi que l'ordre dont lequel sont générées les alertes. De plus, le fait de mettre à jour le degré de gravité associé à l'ensemble des alertes permet de mettre à jour la base de connaissances concernant l'attaque. Ainsi, si la valeur du niveau de gravité de l'ensemble des alertes est positive, c'est-à-dire que l'ensemble des alertes est observé dans un certain ordre, alors l'événement ou l'attaque que décrivent ces alertes peut être dangereux. Par contre, si la valeur est négative, la situation est moins dangereuse. Lorsque la valeur de gravité est nulle, l'alerte dupliquée est ignorée.

La deuxième technique d'agrégation (aggregation relationship) consiste à regrouper les alertes ayant entre elles certaines caractéristiques communes. Pour regrouper les alertes entre elles, l'algorithme AC impose donc des conditions sur leurs attributs appelées *situations*. Une situation est définie par quatre termes :

- **Classe d'alerte :** ce premier terme indique que si une classe d'alerte est spécifiée, alors toutes les alertes agrégées avec cette alerte auront la même classe.
- **Source :** indique l'adresse source que les alertes agrégées ont en commun.
- **Destination :** spécifie l'adresse cible commune à toutes les alertes agrégées.
- **Niveau de sévérité :** ce terme représente un seuil. Si le niveau de gravité de l'ensemble des alertes agrégées dépasse ce seuil, alors une alerte est générée.

Lorsque un ou l'ensemble des termes classe d'alerte (C), la source (S) et la destination (D) sont spécifiés, alors sept différentes situations sont possibles (voir [Debar & Wespi 2001]). Par exemple, la *Situation 1* regroupe les alertes ayant la même source, même destination et même classe d'alerte. Ce type de situations permet par exemple la détection d'un ordinateur attaquant plusieurs serveurs Web.

L'algorithme de corrélation et d'agrégation d'alertes (AC) proposé dans [Debar & Wespi 2001] est utilisé pour regrouper et corréler les conséquences des alertes dans le but de réduire le nombre d'alertes générées et construire des scénarios d'attaques.

Modèle M2D2

Dans [Morin *et al.* 2002], un modèle relationnel de données pour la corrélation d'alertes, nommé M2D2 a été proposé. Quatre différents types d'informations jugés nécessaires à la corrélation d'alertes sont définis :

- Informations sur le système d'informations surveillé. Ces informations spécifient la cartographie du système [Morin 2004].
- Informations sur des attaques et des vulnérabilités.
- Informations concernant la configuration des outils de sécurité.
- Informations sur les événements, en particulier sur les alertes générées.

Les informations concernant la cartographie du système renseignent sur les différentes propriétés qui regroupent à la fois la topologie du réseau (entités physiques comme les hôtes, routeurs et leurs interconnexions), les composants logiciels (systèmes d'exploitation, services) ainsi que leur configuration. La topologie sur laquelle se base le modèle M2D2 est inspirée de celle de Vigna [Vigna 2003]. Elle forme un hypergraphe[5] où les nœuds correspondent à des interfaces du réseau. L'ensemble des arêtes est constitué de deux sous ensembles (hôtes et liens) qui forment chacun une partition de l'ensemble des interfaces. Une fois la topologie physique du réseau est spécifiée, la topologie logique est obtenue en définissant une fonction appelée injection **addr** qui associe une adresse IP à chaque interface. Quelques extensions ont été apportées au modèle M2D2 comme les associations entre les noms des hôtes et adresses IP. Les éléments logiciels (produits) sont formellement modélisés par un quadruplet (*vondor_id,product_id,version_id,type*). Les champs *vondor_id*, *product_id* représentent respectivement le nom de l'entreprise qui développe le produit et le nom du produit. Un nom est associé au produit par la fonction **prodname**. A chaque produit est associé une version obtenue par la fonction **version**. Et finalement, à chaque produit est associé un type, fourni par la fonction **prototype**. Le type d'un produit est de la forme *(OperatingSystem,xxxServer,LocalApp,other)* où *OperatingSystem* spécifie les systèmes d'exploitation, *xxxServer* représente n'importe quel logiciel de type serveur en écoute sur le réseau. xxx doit être remplacé par le nom du service correspondant tels que http, ftp, etc. *LocalApp* spécifie les produits avec lesquels uniquement les relations locales sont possibles.

Les vulnérabilités que le modèle M2D2 considère sont celles référencées dans la liste CVE (Common Vulnerabilities and Exposures) [Mann & Christey 1999] fournie par ICAT[6]. ICAT est une base de données qui utilise les noms des vulnérabilités CVE/CAN pour identifier les vulnérabilités. Du fait que le modèle M2D2 ne contient que les descriptions des vulnérabilités CVE, une fonction notée **equiv** a été définie pour fournir l'ensemble des équivalences entre les noms CVE et les noms des autres systèmes de nommage tels que les identifiants Nessus [Deraison 2000] et Xforce[7]. De plus, les conditions nécessaires à l'exploitation des vulnérabilités sont modélisées par trois classes d'accès : *Remote*, *RemoteUser* et *Local*. Les conséquences de l'exploitation des vulnérabilités sont modélisées par l'utilisation de trois classes : DoS (dénis de service), InformationGathering (collecte d'informations) et CodeExecution (exécution de code).
Les outils de sécurité inclus dans le modèle M2D2 consistent à surveiller le système, détecter des vulnérabilités, etc. Ils sont modélisés par leur type (fonction **type**) comme des sondes de détection d'intrusions (HIDS, NIDS), des scanneurs de vulnérabilités et des pare-feux. Les sondes réseaux utilisent la méthode d'analyse par signatures ou l'approche comportementale pour détecter les intrusions (la méthode d'analyse utilisée est indiquée par la fonction **meth**). Les scanneurs de vulnérabilités testent l'existence de vulnérabilités sur un ensemble de hôtes (en utilisant la relation **scan**). Les pare-feux sont utilisés pour empêcher des connexions indésirables dans un réseau.
Dans le cadre du modèle M2D2, Morin et al [Morin *et al.* 2002] considèrent une alerte générée par un IDS ou par un scanneur de vulnérabilité comme un type particulier d'événement. M2D2 modélise les alertes, les scans et les événements. Les événements qui sont modélisés dans M2D2 sont principalement des événements réseaux (IP, TCP, UDP, HTTP), mais d'autres types peuvent être aussi ajoutés par la relation d'héritage. Les relations qui lient les événements et les alertes sont modélisées par la relation **causes**. Ainsi, une alerte peut être liée à plusieurs

[5] Un hypergraphe est un graphe dont les arêtes ne relient plus un ou deux sommets, mais un nombre quelconque de sommets.
[6] http ://icat.nist.gov
[7] http ://xforce.iss.net

événements mais il se peut que plusieurs alertes peuvent être les causes d'un seul événement comme l'illustre l'exemple qu'on trouve dans [Morin *et al.* 2002] : certains IDSs peuvent générer trois alertes pour la requête "GET /cgi-bin/phf?/etc/passwd" : une concerne la présence de /etc/passwd dans l'URL, une relative à la présence de /cgi-bin/phf et une autre relative au succès ou l'échec de la requête. Une alerte peut également n'être liée à aucun événement, ce qui ne signifie pas que l'alerte n'a aucune cause.

La corrélation d'alertes dans le cadre du modèle M2D2 propose plusieurs techniques d'agrégation selon plusieurs critères. La première technique consiste à regrouper des alertes ayant le même hôte destination. La seconde technique se base sur un ensemble d'informations pour identifier des hôtes vulnérables à l'exécution d'une ou plusieurs attaques. La dernière technique consiste à détecter les faux positifs. Cette technique s'avère intéressante du fait qu'elle permet de prendre en considération plusieurs types d'informations.

Les deux approches [Debar & Wespi 2001] et [Morin *et al.* 2002] que nous avons présentées dans cette section sont classées dans une catégorie de corrélation explicite qui se base sur les scénarios d'attaques connues pour reconnaître les liens entre les alertes. D'autres approches existent comme celles proposées dans [Habra *et al.* 1992, Morin & Debar 2003].

5.4.3 Approches basées sur les pré-conditions et les post-conditions des actions

Les pré-conditions sont des conditions ou pré-requis nécessaires devant être satisfaits sur le système pour que les attaques visant ce système puissent être réalisées. Les post-conditions sont les effets ou conséquences de l'exécution des attaques sur le système ciblé. Ce type d'approches appartient à la catégorie de la corrélation semi-explicite où la fonction principale consiste à reconnaître les plans d'attaques et découvrir des liens de corrélation entre attaques.

Dans [Cuppens & Miège 2002], les auteurs ont proposé une technique de corrélation d'alertes en utilisant le langage LAMBDA [Cuppens & Ortalo 2000]. LAMBDA permet de spécifier les pré-requis et les conséquences des attaques. L'idée de leur approche consiste à spécifier les liens logiques possibles entre les post-conditions d'une attaque A et les pré-conditions d'une autre attaque B. Si de tels liens existent, alors il est possible de corréler une occurrence de l'attaque A avec une occurrence de l'attaque B. Reprenons l'exemple cité dans [Cuppens & Miège 2002] pour montrer ce type de corrélation : Soit deux attaques "MIR-0163" qui consiste à monter les partitions NFS (NFS mount) et "MIR-0164" qui consiste à modifier le fichier ".rhost". Une des post-conditions de "MIR-0163" est post("MIR-0163") = *can-access(Source-user1,Partition1)*. Une des pré-conditions de "MIR-0164" est pre("MIR-0164") = *can-access(Source-user2,Partition2)*. Les attaques "MIR-0163" et "MIR-0163" sont donc directement corrélables car les post-conditions de "MIR-0163" et les pré-conditions de "MIR-0164" sont unifiables. Pour les attaques qui ne peuvent pas être corrélées directement, les auteurs ont introduit l'idée basée sur le fait que la conséquence ou l'effet d'une attaque peut correspondre à un gain de connaissances de l'attaquant sur sa cible. Pour cela, ils ont utilisé un prédicat *knows* qui permet d'exprimer une telle connaissance : une attaque dont une de ses post-conditions est $knows(C)$ est possible qu'elle soit corrélable avec une autre attaque si l'une de ses pré-conditions est C. De plus, les auteurs ont proposé d'introduire des règles ontologiques fournies par un expert pour représenter les relations possibles entre les prédicats. Ces règles se présentent également par des pré-conditions et post-conditions.

Dans [Ning *et al.* 2002], les auteurs ont proposé une approche similaire à celle proposée dans [Cuppens & Miège 2002]. Leur approche se base également sur la modélisation des actions exécutées par l'attaquant par leurs pré-requis et leurs conséquences sur le système. Ces pré-requis et conséquences sont exprimés par des prédicats logiques. Ils ont également introduit la notion d'hyper-alerte pour représenter les alertes. Dans [Templeton & Levitt 2000], les auteurs ont défini un langage de description d'attaques *JIGSAW* permettant de prendre en considération des pré-requis et conséquences des attaques afin de pouvoir les décrire. Deux utilisations de ce langage sont possibles, la première concerne la découverte de scénarios à partir d'un ensemble d'attaques et la seconde concerne la reconnaissance de scénarios d'attaques en temps réel.

En général, l'idée principale des approches appartenant à cette catégorie de corrélation semi-explicite consiste en général à spécifier uniquement les pré-conditions et les post-conditions des différentes attaques afin de générer automatiquement les différents scénarios. L'avantage de ces approches est que l'administrateur n'est pas chargé de décrire tous les scénarios comme dans les approches explicites, mais la spécification de toutes les pré-conditions et les post-conditions des attaques reste toutefois une tâche assez difficile vu le nombre énorme de scénarios.

5.5 Conclusion

Dans ce chapitre, nous avons présenté quelques principes de base de la détection d'intrusions et de la corrélation d'alertes qui sont des mécanismes indispensables pour la sécurité des systèmes d'informations. Nous avons présenté, dans un premier temps, la détection d'intrusions en général, les systèmes de détection d'intrusions ainsi que les deux approches principales de la détection (approche comportementale et approche par signatures). Ces deux approches offrent plusieurs avantages pour la sécurité des systèmes surveillés. Cependant leur utilisation rend la tâche de l'administrateur difficile du fait qu'elles génèrent des quantités énormes d'alertes. C'est pourquoi, nous avons présenté dans la suite de ce chapitre, la corrélation d'alertes qui est considérée comme une solution visant à remédier aux problèmes des approches de détection. Dans ce contexte, plusieurs approches de corrélation d'alertes ont été proposées (implicites, explicites et semi-explicites). Chacune de ces approches se base sur un modèle bien particulier : similarités entre attributs, spécification des scénarios d'attaques, définition des pré-conditions et des post-conditions, etc. Mais toutes ces approches visent le même objectif qui consiste en général à réduire le nombre d'alertes produites par les IDSs.

A notre connaissance, aucune approche de corrélation d'alertes ne tient en compte des connaissances et des préférences de l'administrateur réseau, alors que les décisions concernant le type et le nombre d'alertes à analyser revient à l'administrateur. C'est pourquoi nous avons développé une approche de corrélation d'alertes qui permet de prendre en considération les connaissances et les préférences de l'administrateur réseau.

Dans les chapitres qui suivent, nous décrivons nos contributions concernant les deux points abordés dans ce chapitre : la détection d'intrusions et la corrélation d'alertes, où nous utilisons principalement des techniques d'intelligence artificielle (modèles graphiques probabilistes, raisonnement en présence de préférences). Nous présentons d'abord nos contributions concernant la détection d'intrusions où nous utilisons les réseaux bayésiens pour analyser les données réseaux. Nous présentons, par la suite, notre approche de corrélation d'alertes qui se base sur l'une des

logiques de représentation des préférences que nous avons présentée dans les chapitre 3 et 4.

Chapitre 6

Modèles graphiques probabilistes : Réseaux Bayésiens

Sommaire

6.1 Introduction

Les modèles graphiques sont des formalismes efficaces et puissants pour la représentation et l'extraction de connaissances, notamment en présence des informations incertaines, complexes, imprécises ou manquantes. Ils sont utilisés dans de nombreux domaines : en représentation des préférences tels que les réseaux de préférences conditionnelles (CP-nets) [Boutilier *et al.* 2004], dans les problèmes de décisions tels que les diagrammes d'influences [Shachter 1986, Qi & Poole 1995] et les graphes conceptuels [Chein & laure Mugnier 1992], dans les problèmes de classification tels les arbres de décisions [Quinlan], dans les problèmes de satisfaction de contraintes[8] [Montanari 1974] et dans beaucoup d'autres problèmes de raisonnement probabiliste [Jaeger 2004], possibiliste

[8] CSP pour Constraint Satisfaction Problem

[Borgelt *et al.* 2002, Benferhat & Smaoui 2007], etc. Les modèles graphiques sont également utilisés pour le problème de détection d'intrusions [Benferhat & Tabia 2008b, Kruegel *et al.* 2003, Ben Amor *et al.* 2004]. Leur application s'est avérée intéressante et prometteuse car ce problème peut être considéré comme un problème de classification. Il s'agit de décider de la nature des événements réseaux qui sont, soit des comportements normaux, soit anormaux.

Dans ce chapitre, nous nous intéressons particulièrement aux réseaux bayésiens qui sont parmi les modèles graphiques les plus utilisés dans la pratique. Nous donnons, dans un premier temps, une brève description des modèles graphiques. Nous rappelons ensuite les notions de base sur les réseaux bayésiens, leur utilisation dans le domaine de la détection d'intrusions est présentée dans la section 6.4.

6.2 Modèles graphiques

L'aspect fondamental des modèles graphiques est la *modularité* dans le sens où un système complexe est construit par la combinaison de parties plus simples. Deux composantes sont généralement introduites lorsqu'un problème donné est modélisé par un modèle graphique :

- *la première composante graphique* qui permet de représenter le problème traité sous forme d'un graphe qui met en évidence les variables du domaine et les relations d'influences qu'elles entretiennent entre elles, et
- *une autre composante* dépendante du problème modélisé consiste à assurer la cohérence de la structure du graphe construit. En effet, cette composante peut concerner une quantification des paramètres avec des probabilités, des possibilités ou autres, comme elle peut concerner d'autres contraintes telles que les préférences (tables de préférences conditionnelles), contraintes temporelles, mesures de désirabilité (utilité espérée), etc.

6.2.1 Structure d'un modèle graphique

La structure d'un modèle graphique dépend de la nature du problème modélisé, particulièrement en ce qui concerne le type des relations qu'entretiennent les variables du graphe entre elles. Étant donné un ensemble fini de variables aléatoires $X = \{X_1, X_2, ..., X_n\}$, E un sous-ensemble du produit cartésien $X \times X$. $\mathcal{D}_{X_i}$ est un domaine associé à la variable X_i (valeurs possibles que cette variable peut prendre). $G = (X, E)$ est un graphe représentant la structure du modèle graphique utilisé pour modéliser un problème donné (G peut être un réseau bayésien, un diagramme d'influence, un graphe de préférences conditionnelles, etc.). X constitue l'ensemble des nœuds du graphe et E est l'ensemble des liens entre les nœuds dans X. Selon la nature des liens entre les nœuds, il existe deux familles de graphes :

- **Graphes orientés :** un graphe orienté est constitué d'arcs, c'est-à-dire que l'ensemble des liens entre les nœuds du graphe sont orientés. Dans ce type de graphes, il existe la notion de parents et fils : s'il existe un lien orienté entre X_i et X_j ($X_i \rightarrow X_j$), X_i est appelé *parent* de X_j et X_j est le fils de X_i. Un graphe orienté ne possédant pas de cycles (acyclique) est appelé *DAG* (Directed Acyclic Graph). En particulier, un réseau bayésien est un graphe orienté sans cycles.

- **Graphes non orientés :** ces graphes sont également appelés *graphes de markov*. Dans un graphe non orienté, les liens entre les nœuds ne sont pas orientés, c'est-à-dire que

les notions de fils et parents n'existent pas. L'avantage de ce type de graphes est la possibilité de former des cycles, ce qui n'est pas le cas pour les graphes orientés.

6.2.2 Utilisation des modèles graphiques

Les modèles graphiques peuvent être utilisés dans de nombreux domaines d'applications. Le choix d'un modèle graphique dépend de la nature des informations du problème à traiter : recherche d'un plus court chemin, satisfiabilité d'une contrainte, construction de règles ou de modèles, classification d'un événement dans une classe prédéfinie, recherche de la meilleure décision. En particulier,

- **La représentation et extraction des connaissances :** en présence des quantités importantes d'informations variées et complexes, les modèles graphiques apportent certainement beaucoup de facilités pour la représentation et l'extraction de ces informations. Dans le domaine médical par exemple, ce type de représentation est efficace car les informations médicales sont souvent disponibles en grande quantité. Les connaissances contenues dans ces données peuvent être extraites et modélisées par un réseau Bayésien par exemple basé les variables aléatoires du problème (Age, maladie, poids, etc.).
- **La représentation des préférences :** la notion de préférences est introduite lorsqu'un agent se retrouve face à un choix parmi un ensemble d'alternatives, c'est-à-dire qu'il doit réagir sur la base des informations disponibles plus ou moins connues. L'utilisation d'un modèle graphique pour représenter certains types de préférences telles que les préférences conditionnelles [Boutilier 1994] est possible et c'est avérée intéressante vu le nombre d'alternatives qui peut croître exponentiellement avec la taille du problème. Si nous considérons les graphes de préférences conditionnelles (CP-nets) [Boutilier *et al.* 2004], le graphe (orienté) est construit comme suit : considérer les variables sur lesquelles portent les préférences d'un agent comme *les nœuds du graphe* et les relations de dépendances préférentielles entre les variables sont représentées par des *arcs orientés*. Chaque nœud est associé d'une table de préférences conditionnelles. L'objectif de cette représentation est de déterminer la meilleure alternative en parcourant le graphe du haut (nœud racine) en bas (nœuds feuilles). (Voir section 1.4.2 pour plus de détails).
- **L'aide à la décision :** pour les problèmes de décisions, les informations sont souvent incertaines. C'est pourquoi, les modèles graphiques apportent beaucoup de facilités au niveau des calculs de l'incertitude mais également au niveau de l'expression du raisonnement humain (préférences, croyances). A titre d'exemple, certains modèles graphiques tels que les diagrammes d'influences [Shachter 1986, Qi & Poole 1995, Jensen 2001] ne permettent pas uniquement d'évaluer les états du système par des probabilités, mais ils donnent également la possibilité d'associer une valeur de désirabilité (utilité) à chaque décision possible dans le but d'évaluer la qualité de toutes les décisions. D'autres mesures peuvent également être considérées tel que l'aspect temporel dans les processus de décisions markoviens [Puterman 2005], etc.
  En détection d'intrusions par exemple, il est question de décider dans quelle catégorie faut-il classer un événement ayant lieu dans le réseau. C'est pourquoi les réseaux bayésiens sont des modèles graphiques qui répondent parfaitement à ce type de problèmes essentiellement par leur structure graphique et leur aspect algorithmique comme l'apprentissage automatique, l'inférence et la classification.
- **L'apprentissage automatique :** pour construire un modèle graphique permettant de modéliser un problème particulier, il est nécessaire de formaliser et représenter des connais-

sances qu'on a sur ce problème. Ces connaissances sont fournies soit par un expert, soit automatiquement à l'aide d'outils d'apprentissage automatique. L'apprentissage automatique consiste à construire à partir des données disponibles qui sont souvent en grandes quantités des représentations compactes, et compréhensibles, sous forme de règles, modèles graphiques, etc.

Dans la suite de ce chapitre, nous nous intéressons aux réseaux bayésiens qui sont des modèles graphiques probabilistes.

6.3 Réseaux bayésiens

Les réseaux bayésiens [Jensen 1996, Pearl 1988, Naïm *et al.* 2007], également appelés réseaux probabilistes, sont des outils de modélisation de connaissances incertaines et complexes. Ils permettent aussi la représentation des relations d'influences entre ces connaissances. Ils ont été utilisés dans de nombreuses applications tels que dans le diagnostic médical [François & Leray 2004], en bioinformatique [Wilkinson 2007], corrélation d'alertes [Qin 2005, Benferhat *et al.* 2008], reconnaissance de la parole [Zhou *et al.* 2005], filtrage de spams [Sahami *et al.* 1998], détection d'intrusions [Benferhat & Tabia 2008b],
[Kruegel *et al.* 2003] [Ben Amor *et al.* 2004], etc.

Ces modèles de raisonnement probabiliste se caractérisent par deux aspects : un aspect *graphique* ou qualitatif permettant de représenter d'une manière très simple la connaissance sous forme d'un graphe orienté sans cycles, et un aspect *probabiliste* ou quantitatif offrant un moyen de quantifier l'incertitude des relations d'influences entre les variables du domaine étudié. Plus précisément, un réseau bayésien est défini comme un graphe orienté acyclique (DAG) permettant de représenter les dépendances directes ou conditionnelles entre les variables du domaine étudié. Il est muni d'un ensemble de tables de probabilités conditionnelles (CPT) pour quantifier l'incertitude relative aux relations d'influences.
Formellement, étant donné un ensemble de variables aléatoires $X = \{X_1, X_2, ..., X_n\}$,
$\beta =< G, \Theta >$ est un réseau bayésien où $G = (X, E)$ est un graphe orienté acyclique représentant la structure graphique de β. X est l'ensemble des nœuds où chacun représente une variable aléatoire X_i. A chaque X_i, on associe une table de probabilités locales θ_i qui représente les probabilités des valeurs de X_i sachant toutes les valeurs possibles de leurs parents. E est l'ensemble des arcs représentant les relations de dépendance directe entre les différents nœuds du graphe G.

Les réseaux bayésiens permettent de représenter d'une manière compacte une distribution de probabilités jointe associée à l'ensemble des variables en utilisant la notion d'indépendance. Une distribution de probabilité jointe sur n variables binaires est composée de 2^2 entrées. La distribution de probabilités jointe est décomposée sous forme d'un produit des distributions de probabilités locales selon la règle de chaînage.

$$P(X_1, X_2, ..., X_n) = \prod_{i=1...n} P(X_i | Pa(X_i)), \tag{6.1}$$

où $Pa(X_i)$ représente l'ensemble des parents de X_i. La probabilité conditionnelle d'une valeur d'une variable X_i sachant la valeur d'une autre variable X_j peut être calculée par la loi de *Bayes* de la manière suivante :

$$P(X_i|X_j) = \frac{P(X_j|X_i).P(X_i)}{P(X_j)} \tag{6.2}$$

Les distributions de probabilités locales doivent satisfaire les conditions de normalisation :

- Si X_i est un nœud sans parent du réseau bayésien alors la distribution locale associée à X_i doit satisfaire la condition suivante :

$$\forall x_i \in \mathcal{D}_{X_i}, \sum_{x_i} P(x_i) = 1. \tag{6.3}$$

- Si X_i a des parents dans le réseau bayésien, alors la distribution conditionnelle associée à X_i doit satisfaire :

$$\sum_{x_i} P(x_i|Pa(X_i)) = 1 \tag{6.4}$$

Exemple 6.1

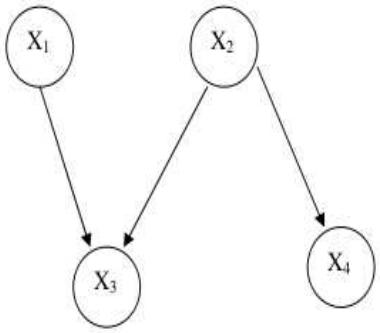

FIG. 6.1: Exemple d'un réseau bayésien

Les variables aléatoires de ce problème sont $\{X_1, X_2, X_3, X_4\}$. Les domaines de ces variables sont : $\mathcal{D}_{X_1} = \{x_1, \neg x_1\}$, $\mathcal{D}_{X_1} = \{x_2, \neg x_2\}$, $\mathcal{D}_{X_3} = \{x_3, \neg x_3\}$, $\mathcal{D}_{X_4} = \{x_4, \neg x_4\}$.
Les distributions de probabilités locales associées aux variables X_1,X_2, X_3,X_4 sont données dans la tableau suivant :

X_1	$P(X_1)$
x_1	0.3
$\neg x_1$	0.7

X_2	P(X_2)
x_2	0.6
$\neg x_2$	0.4

X_4	P($X_4|x_2$)	P($X_4|\neg x_2$)
x_4	0.9	0.35
$\neg x_4$	0.1	0.65

X_3	$P(X_3|x_1x_2)$	$P(X_3|x_1\neg x_2)$	$P(X_3|\neg x_1x_2)$	$P(X_3|\neg x_1\neg x_2)$
x_3	1	1	0.8	0.25
$\neg x_3$	0	0	0.2	0.75

TAB. 6.1: Les distributions de probabilités locales

La distribution de probabilité jointe globale associée au réseau bayésien de cet exemple est :
$\mathrm{p}(X_1, X_2, X_3, X_4) = \mathrm{p}(X_1).\mathrm{p}(X_2).\mathrm{p}(X_3 \mid X_1, X_2).\mathrm{p}(X_4|X_2)$.

6.3.1 Apprentissage des réseaux bayésiens

L'apprentissage d'un réseau bayésien consiste à définir la structure graphique et associer des tables de probabilités conditionnelles à chaque variable du problème à modéliser. Il s'agit donc d'apprentissage de structures et de paramètres.

1. **Apprentissage de structures :** l'apprentissage de structure d'un réseau bayésien consiste à identifier les nœuds et les relations possibles entre ces nœuds à partir des données d'apprentissage. La recherche de structure de réseaux bayésiens est un problème difficile [Chickering 1996], principalement à cause du fait que l'espace de recherche est exponentiel en fonction du nombre de variables décrivant le domaine. C'est pourquoi, de nombreux algorithmes d'apprentissage automatique ont été proposés : algorithme de recherche de causalité [Spirtes *et al.* 2001], algorithme de poids minimum [Chow & Liu 1968], l'algorithme K2 [Cooper & Herskovits 1992], consistent k-graphs [Carvalho & Oliveira 2007], etc. La recherche de la meilleure structure d'un réseau bayésien peut se faire par exemple en parcourant tous les graphes possibles, de leur associer une valeur quantitative (score), puis de choisir le graphe ayant le score le plus élevé [Cooper & Herskovits 1992].

2. **Apprentissage de paramètres :** l'apprentissage de paramètres d'un réseau bayésien consiste à associer une table de probabilités locales à chaque nœud de la structure du réseau préalablement élaborée ou apprise. Les paramètres peuvent être donnés directement par l'expert (ses connaissances subjectives) ou calculés à partir de données d'apprentissage. En présence d'un ensemble de données d'apprentissage et de la structure du réseau, il est simple de calculer les probabilités conditionnelles. Le calcul peut se faire de deux manières différentes selon la nature des données (données complètes ou incomplètes) :
   - Concernant le cas où toutes les données sont complètes (données observées), les probabilités sont calculées sur la base des fréquences qui représentent le nombre d'apparitions de chacune des valeurs que le nœud peut prendre. Il s'agit dans ce cas d'un apprentissage statistique basé sur le *maximum de vraisemblance*. D'autres méthodes peuvent être également utilisées. Nous citons par exemple des méthodes qui se basent sur des estimations bayésiennes (maximum a posteriori et espérance a posteriori).
   - Lorsque les données sont incomplètes, c'est-à-dire que les variables sont complètement manquantes ou ne sont observées que partiellement, plusieurs traitements sont possibles selon la nature des données. Nous citons par exemple la méthode basée sur *l'analyse des exemples disponibles* qui consiste à calculer seulement les probabilités des variables X_i et les probabilités de leur parents $Pa(X_i)$. Ainsi, pour estimer $P(X_i|Pa(X_i))$, il suffit d'utiliser tous les exemples où X_i et $Pa(X_i)$ sont complètement observées. Le lecteur intéressé peut consulter [Leray 2006, Heckerman 1999] pour plus d'informations.

6.3.2 Inférence dans les réseaux bayésiens

L'inférence dans un réseau bayésien concerne le calcul de la probabilité de n'importe quelle variable ou sous ensemble de variables à partir des autres variables observées. Il s'agit donc de déterminer les probabilités conditionnelles d'événements reliés par des relations d'influences. Les

algorithmes d'inférence dans les réseaux bayésiens se répartissent en deux groupes : *algorithmes d'inférence exacte* et algorithmes d'inférence approchée. Les algorithmes d'inférence exacte exploitent les indépendances conditionnelles contenues dans le réseau pour calculer les probabilités a posteriori exactes [Pearl 1988, Lauritzen & Spiegelhalter 1988]. Concernant la deuxième catégorie d'algorithmes, les méthodes utilisées donnent des estimations approchées des probabilités a posteriori [Shachter & Peot 1990, Fung & Favero 1994].

6.3.3 Classification dans les réseaux bayésiens

La classification est considérée comme un cas particulier d'inférence : une seule variable, dite variable de classe que nous symbolisons par C et les autres variables notées A_i, constituent les attributs. A partir des valeurs des attributs, la classification rend comme résultat la classe ayant la plus grande probabilité a posteriori $P(c_i/A)$. Comme exemples de classifieurs bayésiens, on trouve : classifieur naïf de Bayes (Naive Bayes) [Friedman *et al.* 1997], classifieur bayésien naïf augmenté TAN (Tree Augmented Naive Bayes)[Keogh & Pazzani 2002], classifieur BAN (Bayesian Network Augmented Naive Bayes) [Friedman *et al.* 1997], les BMN (Multi-Nets Bayesians) [Cheng & Greiner 2001]. Dans les sections qui suivent, nous présentons les classifieurs bayésiens naïf et TAN.

6.3.4 Réseaux bayésiens naïfs

Une variante simple des réseaux bayésiens est appelée réseaux bayésiens naïfs. Ces réseaux sont bien adaptés pour le traitement des problèmes de classification [Friedman & Goldszmidt 1996] du fait que leur structure est unique et fournie a priori par l'expert du domaine. Deux principaux éléments forment la structure d'un réseau bayésien naïf :

- Un seul nœud racine (parent de tous les autres nœuds) représentant la variable de la classe C.
- des nœuds fils, notés A_i représentants les différents attributs.

Comme le montre la figure 6.2, les nœuds fils n'entretiennent entre eux aucune relation d'influence. Les seules relations possibles sont celles entre la classe et les attributs (Classe → attributs).

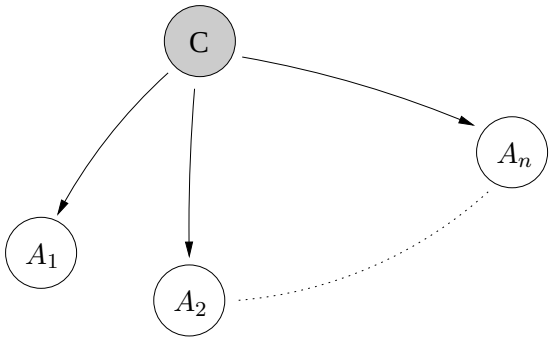

FIG. 6.2: Structure d'un réseau bayésien naïf

La classification d'un nouvel objet (recherche de la classe c_i la plus probable) étant données les valeurs de ses attributs est donnée par la loi de Bayes (adaptation de l'équation 6.2 pour les réseaux bayésiens naïfs).

$$P(c_i|A) = \frac{P(A|c_i).P(c_i)}{P(A)} \tag{6.5}$$

Si a_1, a_2,...a_n sont les valeurs relatives aux attributs A_i, tel que $A = (A_1 = a_1, A_2 = a_2, ..., A_n = a_n)$, l'hypothèse d'indépendance des attributs dans le contexte de la classe nous permet d'écrire :

$$P(c_i|A) = \frac{P(A|c_i).P(c_i)}{P(A)} = \frac{P(A_1 = a_1|c_i).P(A_2 = a_2|c_i).....P(A_n = a_n|c_i).P(c_i)}{P(A)} \tag{6.6}$$

Notons que nous n'avons pas besoin de calculer la probabilité de l'évidence $P(A)$ puisqu'elle est déterminée par la condition de normalisation (elle est la même pour toutes les classes). Donc il est suffisant de calculer pour chaque classe c_i ce qui suit :

$$P(c_i|A) = P(A_1 = a_1|c_i).P(A_2 = a_2|c_i).....P(A_n = a_n|c_i).P(c_i) \tag{6.7}$$

6.3.5 Réseaux bayésiens naïfs augmentés

Pour des problèmes de classification, l'utilisation des réseaux bayésiens naïfs donne généralement des résultats satisfaisants malgré leur simple structure et l'hypothèse d'indépendance conditionnelle qui ignore les dépendances entre attributs. Cependant, ces réseaux ne sont pas toujours adaptables pour certains types de problèmes. C'est pourquoi, Friedman et al [Friedman & Goldszmidt 1996] ont proposé les réseaux bayésiens naïfs augmentés (TAN) dans le but de représenter quelques dépendances entre les attributs. Le nœud de classe est parent de tous les attributs. De plus, dans un TAN, un attribut peut avoir au maximum deux parents : le nœud de classe et éventuellement un autre attribut. La structure obtenue en ignorant les dépendances entre les attributs et les nœuds de classe est un arbre, d'où l'appellation réseau bayésien naïf augmenté d'un arbre. La figure 6.3 montre un exemple d'un TAN.

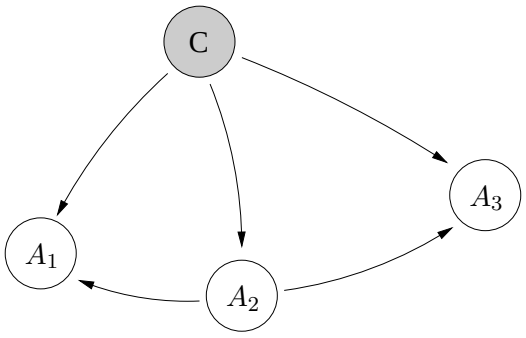

Fig. 6.3: Exemple de structure d'un réseau bayésien naïf augmenté (TAN)

6.4 Application des modèles graphiques pour la détection d'intrusions

Nous avons présenté dans le chapitre précédent deux principales approches que les systèmes de détection d'intrusions adoptent pour la détection d'intrusions : approche comportementale et approche par signatures. Durant ces dernières années, de nouveaux formalismes et techniques sont utilisés pour améliorer la détection et faire face à certains problèmes telle que la complexité du problème de détection due à la nature des données à analyser. En effet, les données qu'un administrateur réseau doit analyser sont souvent hétérogènes (différentes sources), d'une taille très importante et contiennent des informations incertaines, incomplètes, imprécises ou manquantes. Parmi ces nouveaux formalismes, on trouve les modèles graphiques qui par leurs aspects algorithmiques (apprentissage automatique, inférence, etc.) et modulaire ont montré leur efficacité dans le domaine de la détection d'intrusions. Les réseaux bayésiens [Pearl 1988], les réseaux de neurones [Hertz *et al.* 1991] sont parmi les modèles graphiques les plus utilisés en détection d'intrusions. La détection d'intrusions peut être vue ou traitée comme un problème de classification [Lee 1999]. Il s'agit d'analyser les évènements ayant lieu au sein du système d'informations dans le but de les classifier ou les identifier selon la nature des activités qu'ils comportent (normales ou anormales). Cette approche est très prometteuse car les modèles de classification peuvent être construits directement à partir de données représentant des activités normales et anormales. Une fois le modèle construit, il sera utilisé pour classer des nouveaux événements.

6.4.1 Détection d'intrusions et techniques d'apprentissage automatique et classification

Les techniques de classification permettent de déterminer l'appartenance d'un objet à une catégorie prédéfinie. Ces techniques consistent à élaborer sur les données d'apprentissage, un modèle de classification pouvant être généralisé sur l'ensemble des données du problème.
En détection d'intrusions, un objet correspond à un événement qui peut être une connexion ou un paquet caractérisé par un vecteur d'attributs (protocole, durée, flag, service, etc.). La classe de l'événement est soit normale, soit anormale (nom de l'attaque).

Les réseaux bayésiens sont largement utilisés pour ce type de problèmes. Parmi ces travaux, nous citons par exemple les travaux de Lee [Lee 1999] où des techniques de fouilles de données (telles que les règles d'associations) sont utilisées sur la base de données $KDD'99$ [KDD 1999] dans le but de construire des modèles permettant de distinguer entre activités normales et suspectes. Dans [Ben Amor *et al.* 2004], les auteurs ont expérimenté les réseaux bayésiens naïfs [Friedman *et al.* 1997] et les arbres de décisions [Quinlan 1993] sur la base de données $KDD'99$, et ils ont montré que ces deux algorithmes donnent des résultats satisfaisants. Dans [Benferhat & Tabia 2005], un schéma d'hybridation exploitant les performances de chaque algorithme (réseau bayésien naïf et arbre de décisions) a été proposé ayant pour résultat une amélioration significative dans les taux de classification globales et les taux de détection des attaques $R2L$ et $U2R$ sur la base $KDD'99$. Une nouvelle approche basée sur l'apprentissage supervisé afin de détecter des nouvelles attaques a été proposée dans [Bouzida 2006]. Plusieurs d'autres travaux existent dans cette thématique de recherche, nous citons par exemple les travaux suivants : [Benferhat & Tabia 2008a, Xuren & Famei 2006, Chimphlee *et al.* 2006]. Le schéma de la figure 6.4 présente les principales étapes concernant la modélisation du problème de détection d'intrusions auquel nous nous intéressons en utilisant les réseaux bayésiens.

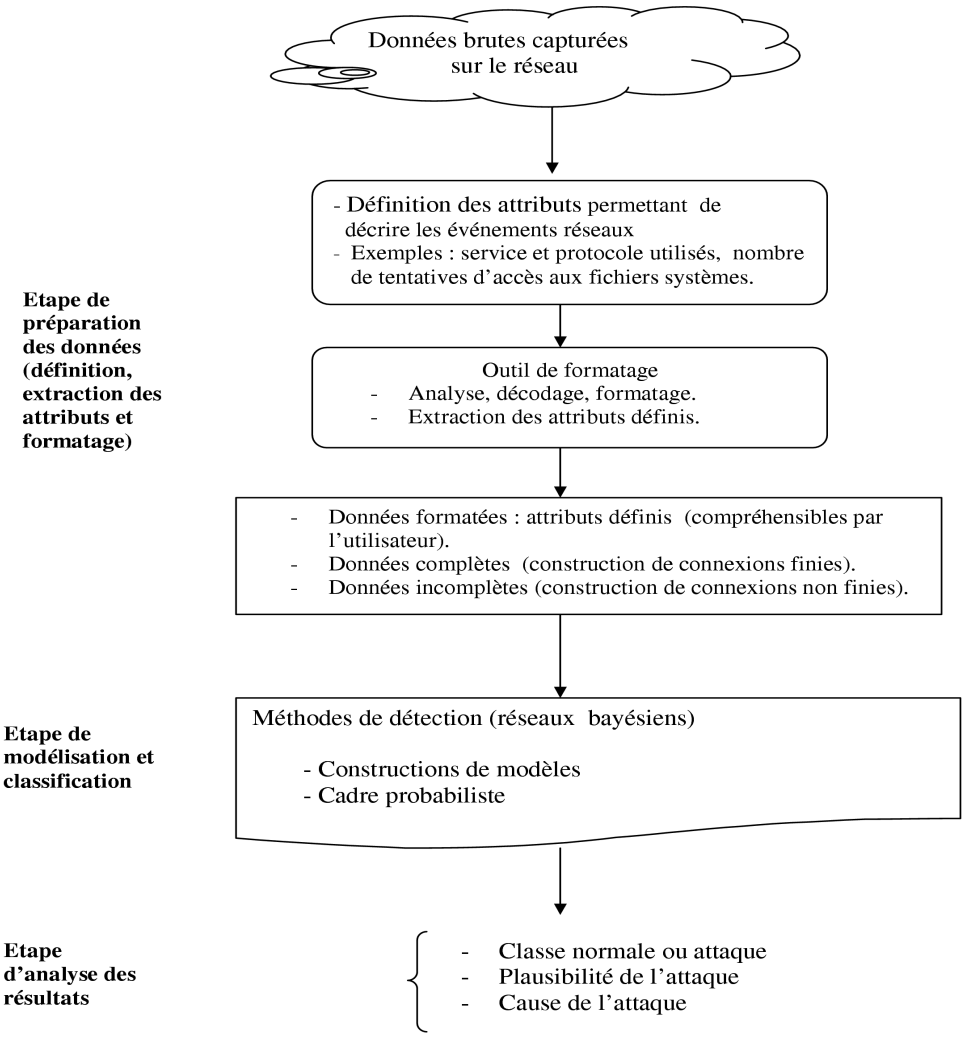

FIG. 6.4: Modélisation d'un problème de détection d'intrusions en utilisant un réseau bayésien

Préparation de données

Lorsque nous nous intéressons à modéliser un problème donné par un modèle graphique, il est nécessaire dans un premier temps de définir les variables du problème afin de pouvoir définir la structure du graphe et les relations d'influences possibles entre ces variables. C'est pourquoi, pour un problème de détection d'intrusions, la première étape de modélisation consiste à extraire les attributs permettant de décrire les différents événements réseaux afin de préparer les données dans un format exploitable par le modèle graphique utilisé et les techniques de détection. Nous présentons cette partie en détails dans le chapitre suivant.

Construction des modèles

Lorsque les données sont fournies dans un format exploitable, des techniques de fouille de données peuvent être appliquées pour extraire les relations de dépendances entre les différents attributs, permettant d'une part de caractériser un comportement spécifique et d'autre part

de les représenter sous forme de règles. Le choix d'une technique d'apprentissage automatique dépend des spécificités du problème traité et de la nature de la solution qu'on veut apporter. La précision, rapidité, nature des données à manipuler (quantité, nombre et type des attributs, nombre de classes, etc.) sont parmi les facteurs qui décident de l'adéquation d'un classifieur à un problème particulier. Il existe des classifieurs très satisfaisants en ce qui concerne le taux de classification ou de détection, mais sont peu adaptables aux applications temps réel où le temps de réponse est critique. Dans d'autres classifieurs, tels que les réseaux bayésiens naïfs, la structure est facile à construire et donnent des résultats satisfaisants, mais exigent beaucoup de données d'apprentissage pour une meilleure estimation de ses paramètres.
La figure 6.5 illustre l'approche générale des techniques de classification :

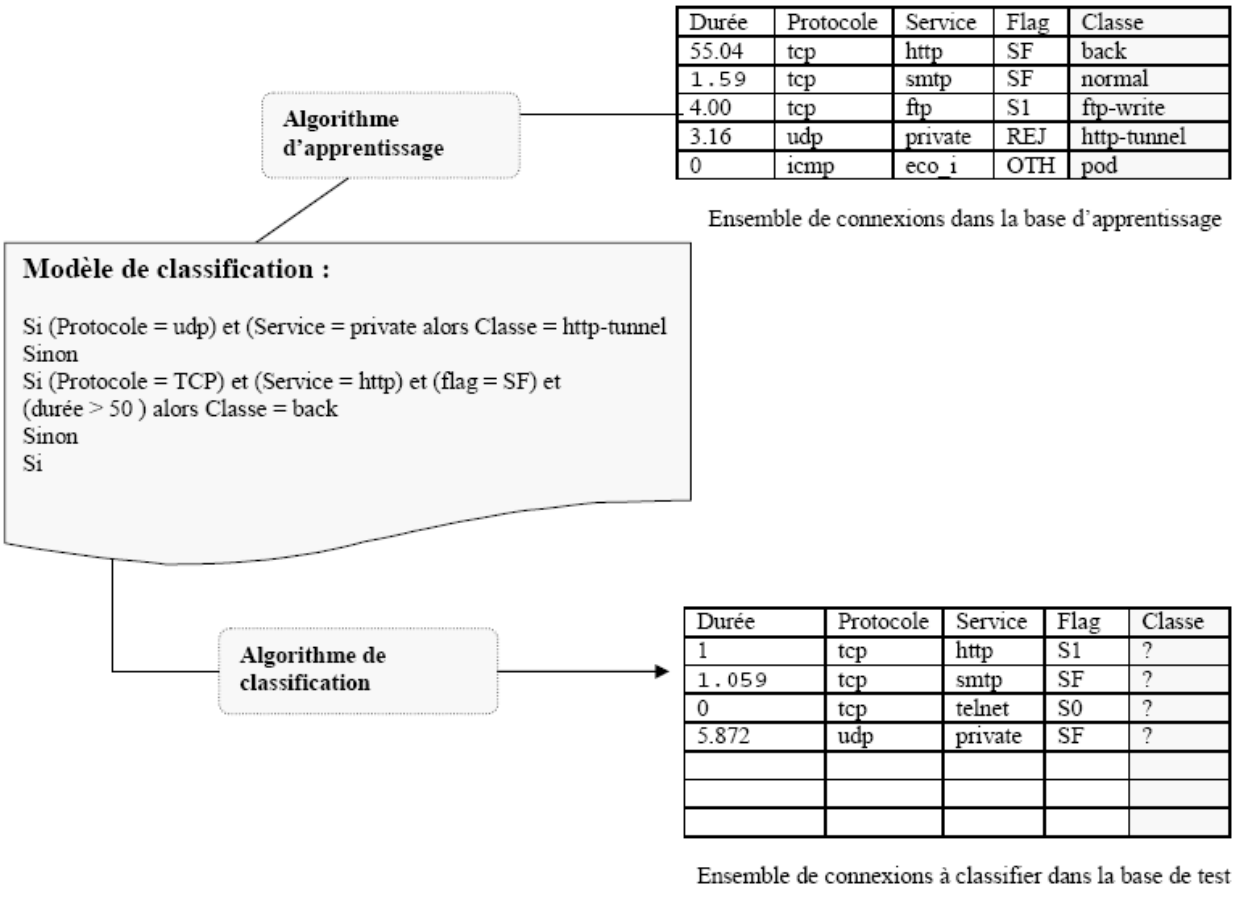

Fig. 6.5: Approche générale de classification

La classification d'objets décrite dans la figure 6.5 est *supervisée*, c'est-à-dire que chaque objet de la base d'apprentissage est associé à une classe. A partir de ces objets, en utilisant un algorithme d'apprentissage, un modèle de classification sera construit et peut être par la suite appliqué pour classifier les objets de la base de test.

Evaluation de la qualité de détection d'intrusions

L'objectif d'un administrateur réseau lorsqu'il envisage de choisir un système de détection d'intrusions est d'assurer au maximum la sécurité du système qu'il surveille. Or, malgré ses connaissances et son expérience sur le fonctionnement de son système, le type d'application et les sources de données qu'il veut surveiller, le choix est souvent difficile à cause du fait qu'il existe plusieurs aspects caractérisant ces IDSs tels que le coût, facilité ou difficulté d'utilisation et d'administration, débit réseau réellement supporté, degré de sécurité, la qualité de détection (notamment le taux de détection et le taux de fausses alertes), etc. Ainsi plusieurs facteurs interviennent pour évaluer un système de détection d'intrusions en terme de performances et limitations. L'évaluation d'un IDS est difficile est coûteuse. C'est la raison pour laquelle, le nombre d'évaluations effectuées à nos jour est faible. Nous citons par exemple les évaluations DARPA 1998 [Lippmann *et al.* 2000] et la comparaison de 10 IDSs dans [Shipley 1999]. Un des critères d'évaluation des IDSs, particulièrement connu ces dernières années, est la courbe ROC (Receiver Operating Characteristic). Ce critère décrit la relation entre deux paramètres de fonctionnement d'un IDS, sa probabilité de détection et sa probabilité de fausses alertes.

Concernant le problème que nous traitons, nous avons opté pour l'utilisation d'un réseau bayésien naïf comme classifieur (voir section 6.3). Nous nous limitons donc aux paramètres quantitatifs permettant d'analyser la qualité de la détection. L'évaluation est basée sur le taux de détection et le taux d'erreurs de détection. Le taux de classification est le pourcentage des connexions correctement classifiées (PCC) par rapport au nombre total de connexions. Le taux d'erreurs de classification est le pourcentage des connexions incorrectement classifiées par rapport à la totalité des connexions (une erreur de détection peut être une fausse alerte ou une intrusion non détectée). Le PCC est calculé comme suit :

$$PCC = \frac{Nombre \quad des \quad instances \quad correctement \quad \text{classifiées}}{Nombre \quad total \quad d'instances \quad \text{classifiées}} * 100 \tag{6.8}$$

D'autres paramètres peuvent être également utilisés. Nous les résumons dans la matrice de confusion présentée dans le tableau 6.2.

		Classe détectée	
		Normale	Attaque
Classe réelle	Normale	Vrai négatif (TN)	Fausse alerte (FP)
	Attaque	Faux négatif (FN)	Vrai positif (TP)

Tab. 6.2: Matrice de confusion.

- Un vrai négatif (true negative) est une activité normale, détectée également normale par l'IDS.
- Un vrai positif est une attaque correctement détectée par l'IDS.
- Une fausse alerte, appelée également faux positif, est une activité normale mais l'IDS l'a détectée comme une attaque.
- Un faux négatif est une activité détectée comme normale par l'IDS alors qu'elle est réellement une attaque.

Lorsque le taux de fausses alertes générées par l'IDS est trop élevé, l'administrateur réseau peut laisser passer des attaques sans les analyser, il devient alors moins vigilant et cela rend l'IDS moins fiable. Un taux de faux négatifs élevé rend l'IDS inefficace du fait qu'il n'arrive pas à détecter la majorité des attaques. Notons également que dans le cas de classification, une fausse alerte ou un faux négatif sont considérés comme une erreur de la classification. Cependant, une erreur de classification peut ne pas être une erreur de détection dans certains cas. A titre d'exemple, si une attaque d'une certaine catégorie est signalée comme une attaque d'une autre catégorie alors il s'agit d'une erreur de classification, mais pas d'une erreur de détection. Il s'agit en effet d'une erreur d'identification des attaques (nom ou catégorie).

6.4.2 Exemple de modélisation d'un problème de détection d'intrusions par un réseau bayésien

L'exemple suivant illustre comment un problème de détection d'intrusions peut être modélisé comme un problème de classification en utilisant un réseau bayésien.

Exemple 6.2
Supposons que nous avons un ensemble de données d'apprentissage données dans le tableau 6.3. Chaque ligne correspond à une connexion décrite par un ensemble d'attributs qui sont : Protocole, Service et Flag (état de la connexion). Notons que les données réseaux sont de nature brute et elles ne sont pas fournies directement dans ce format (comme le format des données présentées dans le tableau 6.3 par exemple). Pour avoir les données dans un tel format, il est nécessaire de les formater pour extraire des attributs et construire des connexions. Ce point sera abordé dans le chapitre suivant.

Protocole	Service	Flag	Classe (C)
tcp	smtp	SF	Anormale
tcp	http	SF	Normale
tcp	smtp	SF	Anormale
udp	private	RSTO	Anormale
udp	domain	SF	Normale
udp	https	SF	Normale
tcp	hostname	S0	Anormale
tcp	hostname	RSTO	Normale
tcp	http	SF	Normale
udp	domain	SF	Normale

Tab. 6.3: Ensemble de données d'apprentissage

Le réseau bayésien naïf correspondant à l'ensemble de données d'apprentissage donné dans le tableau 6.3 est représenté dans la figure 6.6. La variable Classe (C) correspond au nœud racine du graphe et les autres variables (Protocole, Service, Flag) correspondent aux nœuds fils. $\mathcal{D}_{Classe}$ = {Normale, Anormale}, $\mathcal{D}_{Protocole}$ = {tcp, udp}, $\mathcal{D}_{Service}$ = {http, smtp, private, domain, hostname, https}, $\mathcal{D}_{Flag}$ = {SF, S0, RSTO}.

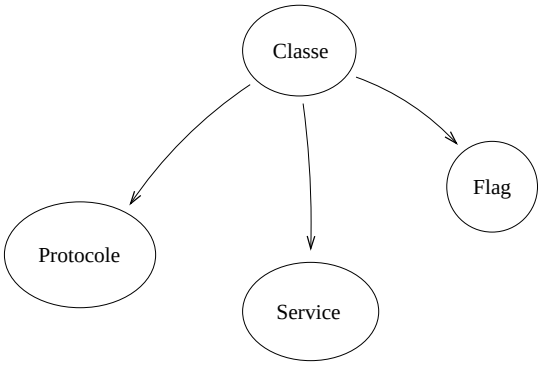

FIG. 6.6: Structure du réseau bayésien naïf de l'exemple

Le calcul des probabilités conditionnelles et a priori se basant sur les fréquences en ajoutant 1 à tous les numérateurs et de compenser ces ajouts dans les dénominateurs (ce type de calcul est important pour éviter les probabilités nulles). Les probabilités conditionnelles et a priori sont représentées dans le tableau suivant :

P(Classe)		P(Protocole | Classe)			P(Service|Classe)		P(Flag|Classe)	
Normale	Anormale		Normale	Anormale	Normale	Anormale	Normale	Anormale
7/12	5/12	tcp	4/8	4/6	http 3/12	1/10	SF 6/9	3/7
		udp	4/8	2/6	smtp 1/12	3/10	S0 1/9	2/7
					private 1/12	2/10	RSTO 2/9	2/7
					domain 3/12	1/10		
					hostname 2/12	2/10		
					https 2/12	1/10		

TAB. 6.4: Probabilités conditionnelles et a priori du réseau de l'exemple

Supposons que nous avons une nouvelle connexion dont on ignore la catégorie. Les attributs Protocole = tcp, Service = smtp et Flag = RSTO décrivent la connexion à classifier.
Pour déterminer la classe de cette nouvelle connexion, nous allons calculer $P(Normale|tcp, smtp, RSTO)$ et $P(Anormale|tcp, smtp, RSTO)$. Par l'application de la loi de Bayes, ces probabilités sont calculées comme suit :

$$P(Normale|tcp, smtp, RSTO) = \frac{P(tcp,smtp,RSTO|Normale).P(Normale)}{P(tcp,smtp,RSTO)}$$

$$P(Anormale|tcp, smtp, RSTO) = \frac{P(tcp,smtp,RSTO|Anormale).P(Anormale)}{P(tcp,smtp,RSTO)}$$

Mais, puisque nous utilisons un réseau bayésien naïf, il est suffisant de calculer pour chaque classe cette probabilité comme suit :

- $P(Normale|tcp, smtp, RSTO) = P(tcp|Normale).P(smtp|Normale).P(RSTO|Normale).P(Normale)$ $= (4/8).(1/12).(2/9).(7/12) = 0.01$,
- $P(Anormale|tcp, smtp, RSTO) = P(tcp|Anormale).P(smtp|Anormale).P(RSTO|Anormale).P(Anormale)$ $= (4/6).(3/10).(2/7).(5/12) = 0.02$.

La classe la plus probable est la classe "Anormale", donc la nouvelle connexion est Anormale.

Nous pouvons également représenter ce problème par un réseau bayésien naïf augmenté comme celui présenté dans la figure 6.7.

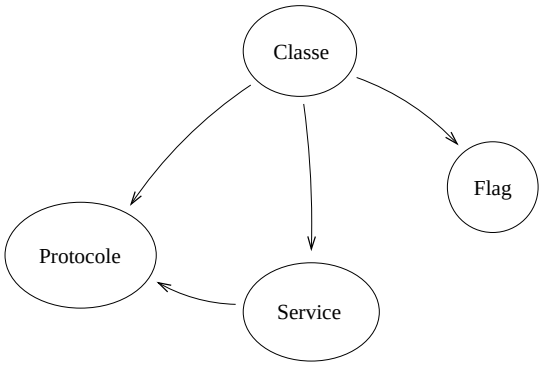

FIG. 6.7: Exemple d'un réseau TAN pour la classification de connexions réseaux

Selon le graphe de la figure 6.7, le réseau montre une relation entre les attributs Service et Protocole (Service $\longrightarrow$ Protocole) car le service (http, smtp, private, domain, hostname, https) détermine le protocole utilisé (tcp, udp). Les probabilités conditionnelles et a priori lorsque nous utilisons un réseau bayésien naïf augmenté sont les mêmes dans le cas d'un réseau bayésien naïf pour les attributs Service, flag et la classe, par contre pour le protocole, nous devons calculer P(Protocole | Classe,Service).

6.5 Conclusion

Nous avons présenté dans ce chapitre une brève description des modèles graphiques et leur utilité dans différents domaines. Nous nous sommes intéressés particulièrement aux réseaux bayésiens qui sont considérés parmi les modèles graphiques les plus utilisés. Nous avons également présenté comment un problème de détection d'intrusions peut être modélisé par ces modèles, notamment pour des besoins de classification.

Concernant le problème de détection d'intrusions, les données traitées sont de nature brute, c'est-à-dire inexploitables directement. En effet, il est impossible de les représenter par un modèle graphique. C'est pourquoi, il est nécessaire, dans un premier temps, de définir des attributs permettant de décrire les événements normaux ou anormaux (comme exemples d'attributs : source et destination de l'événement, durée, service utilisé, nombre de tentatives d'accès à un fichier système, etc.). Une fois les attributs définis, il reste à les extraire. Pour cela, la fonction de formatage sert à formater les données brutes et extraire des attributs afin de fournir des données exploitables par les modèles utilisés pour l'analyse et la classification de ces données. La définition, l'extraction des attributs, le formatage des données réseaux brutes et les résultats expérimentaux seront présentés dans le chapitre suivant.

Chapitre 7

Définition d'attributs, formatage du trafic réseau et résultats expérimentaux

Sommaire

7.1 Introduction

La difficulté majeure du problème de détection d'intrusions est principalement liée à la complexité des données en terme d'hétérogénéité (différentes sources, protocoles, etc.), quantité (plusieurs Giga-octets), qualité (format brut et inexploitable directement par l'analyseur), etc. Dans ce contexte, la nécessité de réduire la quantité importante de données à analyser, de les rendre plus significatives, d'automatiser le processus d'extraction et d'analyse des données requiert plusieurs éléments. En particulier, cela nécessite la définition d'attributs et le développement d'outils d'extraction d'attributs et de formatage, etc. En effet, pour détecter les actions illégales et détecter les événements suspects dans les flux de trafic réseau, qui sont de nature brute, il est indispensable de définir au préalable des attributs permettant de représenter les

caractéristiques du trafic réseau normal et malveillant. Ces attributs peuvent par la suite être utilisés pour construire des modèles d'activités réseaux capables de distinguer entre activités légitimes et activités malveillantes.

Dans ce chapitre, nous nous intéressons à l'application des modèles graphiques probabilistes au problème de la détection d'intrusions. Pour garantir de meilleurs résultats, nos données d'expérimentations doivent donc être fournies dans un format exploitable par ces modèles. Ainsi, ce chapitre présente deux éléments essentiels : la définition d'attributs permettant de décrire les comportements des attaques et du trafic réseau normal et le formatage du trafic réseau brut pour transformer les paquets réseaux en données formatées (structurées) et d'extraire les attributs préalablement définis.

Le reste du chapitre est organisé comme suit : dans la section 7.2, nous abordons le problème de description des données réseaux et la définition d'attributs. Dans la section 7.2.1, nous présentons les différents attributs que nous avons définis pour nos données d'expérimentations. Dans la section 7.3, nous présentons l'outil de formatage que nous avons développé. La section 7.4.1 décrit les données d'expérimentations que nous avons utilisées. La section 7.5 présente les résultats expérimentaux.

7.2 Définition d'attributs

Dans ce chapitre, nous nous intéressons à la modélisation du problème de détection d'intrusions avec un réseau Bayésien. Notons que le problème de détection d'intrusions peut être considéré comme un problème de classification puisqu'il est question de classifier les paquets ou connexions en événements normaux ou attaques. Ainsi, pour élaborer un modèle de classification pour la détection d'intrusions, les données doivent être représentées sous forme d'objets décrits par un ensemble d'attributs. Les objets à analyser peuvent être des paquets ou des connexions. Chaque attribut prend une valeur dans un domaine de valeurs. La classe associée à chaque objet prend une valeur dans un ensemble de valeurs prédéfinies. Ainsi, la modélisation à base d'un modèle graphique probabiliste consiste à représenter chaque attribut par une variable aléatoire et modéliser les dépendances entre les différents attributs. La classification consiste à assigner à tout nouvel objet, décrit par un vecteur d'attributs, la classe correspondante en se basant sur le modèle construit lors de la phase d'apprentissage. Dans notre contexte, un objet correspond à une connexion caractérisée par un vecteur d'attributs (protocole, durée, flag, service, etc.) et chaque connexion appartient à une classe représentant la nature de l'activité qu'elle comporte (normale ou anormale). Notons que notre objectif dans ce chapitre est de détecter les intrusions avec deux types de données : données complètes et données incomplètes. Pour ce faire, il est impératif que les données réseaux brutes que nous utilisons pour nos expérimentations soient structurées comme suit :

- Connexions finies ou complètes : dans ce cas, les attributs décrivant ces connexions prennent des valeurs complètes ou finies. Pour ces connexions, on attend jusqu'à la fin d'une connexion pour extraire les attributs.
- Connexions non finies ou incomplètes : dans ce cas, ces connexions sont décrites par des attributs qui prennent des valeurs incomplètes ou non finies. Pour ces connexions, on n'attend pas la fin d'une connexion pour extraire les attributs.

La figure 7.1 montre les phases de structuration des données réseaux et d'extraction d'attributs.

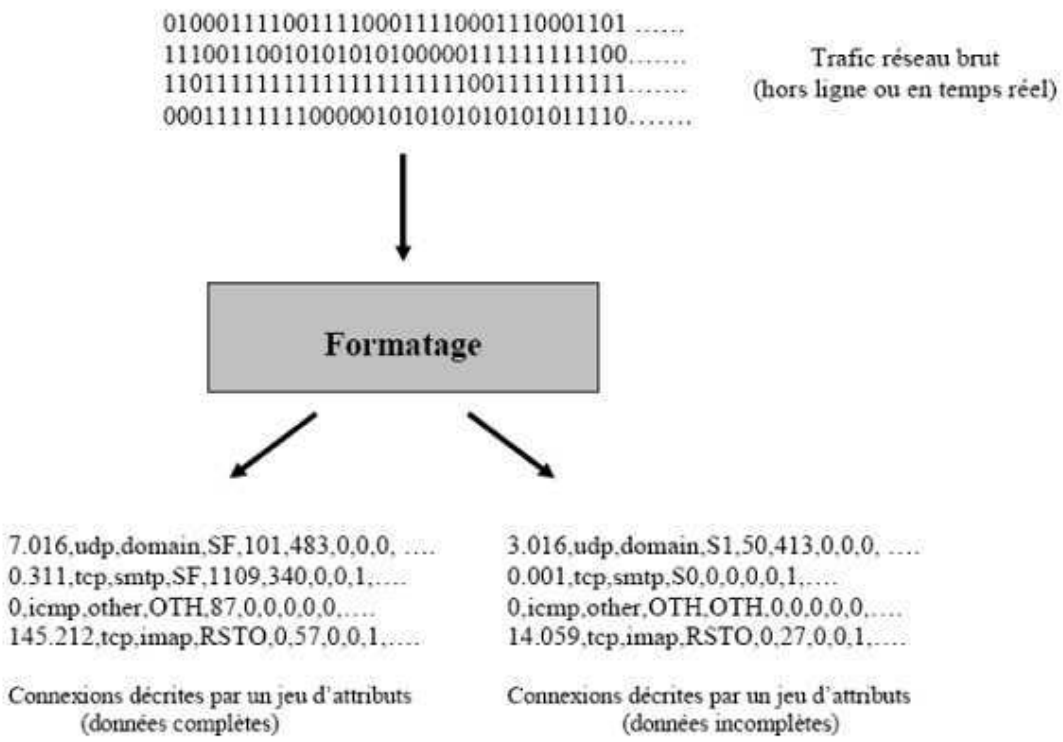

FIG. 7.1: Phase de structuration des données réseaux et d'extraction d'attributs

L'objectif principal de la définition d'attributs pour un problème de détection d'intrusions consiste à proposer un jeu d'attributs garantissant un taux de détection élevé avec un faible taux de fausses alertes. Cette tâche ressemble à celle de l'écriture de signatures dans le sens où les deux ont pour objectif de décrire les données réseaux, les caractéristiques du système surveillé et toute information servant à reconnaître des attaques. Cependant, l'écriture de signatures s'avère une tâche plus délicate car chaque signature doit concerner une attaque particulière, ce qui nécessite des connaissances expertes sur les comportements et les stratégies suivies par des attaques.

7.2.1 Types et natures des attributs définis

Plusieurs types d'attributs peuvent être définis et extraits à partir des données brutes. Certains sont basiques et faciles à extraire (comme l'adresse IP source, service, etc.) et d'autres sont de haut niveau comme des attributs représentant le contenu des paquets (nombre de tentatives d'accès en mode *root* par exemple). Il est également possible d'extraire des attributs relatifs à plusieurs connexions dans le but de caractériser leur historique. Pour cela, deux aspects sont pris en considération : le premier concerne une fenêtre temporelle (2 secondes par exemple) alors que l'autre aspect concerne une fenêtre de connexions (100 dernières connexions par exemple).
Nous distinguons en tout cinq classes d'attributs : les attributs des quatre premières classes sont les mêmes que ceux de la base de données KDD'99 [KDD 1999] alors que ceux de la cinquième classe sont de nouveaux attributs. Notons que tous ces attributs sont extraits par l'outil de formatage que nous avons développé à cet effet [Benferhat *et al.* 2007].

7.2.2 Principales attaques réseaux

Par attaques réseaux, nous entendons les attaques ciblant le réseau lui-même ou une machine du réseau ciblé. Ces attaques génèrent du trafic réseau et peuvent être initiées par un utilisateur interne ou externe. Les principales attaques qui génèrent du trafic réseau sont les suivantes :

1. **Déni de service (DoS) :** ce type d'attaques vise à perturber ou dégrader le fonctionnement normal d'un réseau et ses ressources. Certaines attaques DoS saturent un service ou un réseau avec un grand nombre de paquets alors que d'autres provoquent des dysfonctionnements en envoyant des paquets malformés afin d'exploiter des vulnérabilités et des failles dans certains programmes.

2. **Scans ou probe :** Ces attaques collectent des informations utiles pour l'attaquant (comme les systèmes et ports ouverts) pour lancer par la suite les actions de l'attaque réelle.

3. **Attaques Web :** Ce sont des attaques très fréquentes avec la popularité croissante des services Web. Elles ont pour cibles les applications Web pour provoquer des dénis de services (par exemple, planter un serveur Web), s'accaparer des informations, exécuter des commandes, etc.

Rappelons que notre objectif est de définir des attributs discriminant entre trafic normal et trafic intrusif. Mais puisque nos études expérimentales seront majoritairement réalisées sur la base de données DARPA'99 (présentée plus bas dans ce chapitre), nous nous sommes concentrés sur la définition d'attributs pertinents pour détecter les attaques de cette base de données. De plus, la nature du trafic réseau et des attaques de DARPA'99 ressemble beaucoup à ceux de la base DARPA'98 pour laquelle le jeu d'attributs KDD'99 a été défini. Ainsi, nous avons réutilisé dans nos expérimentations tous les attributs de base, de contenu et d'historique des connexions KDD'99 et nous avons défini d'autres attributs pour enrichir les attributs KDD'99. Ci-après les catégories d'attributs décrivant nos connexions :

1) Attributs de base

Cette classe regroupe neuf attributs intrinsèques qui décrivent les données au niveau paquet [Lee *et al.* 1999]. Par exemple, l'attribut $flag$ donne l'état d'une connexion. Voici certaines valeurs du flag que nous pouvons trouver dans une connexion TCP.

$S0$: Tentative de connexion, mais pas de réponse (connexion demandée).

$S1$: Connexion établie, non terminée.

SF : Établissement et terminaison normale de la connexion.

REJ : Tentative de connexion rejetée.

$S2$: Connexion établie et tentative de fermeture par le hôte source, mais pas de réponse du hôte de destination.

Ces attributs sont faciles à extraire à partir des paquets constituant les connexions mais ils sont importants pour la détection de plusieurs types d'attaques. A titre d'exemple, l'attribut Src_bytes, représentant la quantité d'octets envoyés de la source vers la destination, est indispensable pour détecter les attaques par buffer-overflow. Nous verrons également que certains attributs de cette catégorie (tels que le service, durée, flag, etc.) sont utilisés pour calculer d'autres

attributs (attributs d'historique). Les attributs de base sont présentés dans le tableau 7.1.

Numéro	Nom de l'attribut	Description	Type
(1)	duration	durée de la connexion	numérique
(2)	protocol_type	type du protocole utilisé pour chaque connexion tels que tcp, udp, icmp	symbolique
(3)	service	service de la connexion tels que ssh, http, https, etc.	symbolique
(4)	src_bytes	nombre d'octets transférés de la source vers la destination durant la connexion	numérique
(5)	dst_bytes	nombre d'octets transférés de la destination vers la source durant la connexion	numérique
(6)	flag	état de la connexion tel que S0, REJ, etc.	symbolique
(7)	land	"1" si la source et la destination de la connexion sont les mêmes (adresse IP et port), "0" sinon	symbolique
(8)	wrong_fragment	nombre de fragments erronés dus à la mauvaise fragmentation	numérique
(9)	urgent	nombre de paquets urgents	numérique

Tab. 7.1: Attributs de base des connexions

2) Attributs de contenu

Les attributs appartenant à cette classe sont relatifs aux contenus des paquets d'une connexion. Ces attributs sont très importants car ils permettent de révéler certaines actions malveillantes tels que les accès aux fichiers systèmes, tentatives d'accès non autorisées, etc. Ils sont particulièrement utiles pour la détection des attaques U2R (User to Root) et R2L (Remote to Local access). La définition de ce type d'attributs nécessite des connaissances a priori sur les différentes stratégies d'attaques. Par exemple, au cas où un attaquant tente d'accéder à un fichier "./rhosts", cet accès est considéré sensible, il doit donc être signalé par l'un des attributs (par l'attribut *hot* par exemple). Ces attributs sont présentés dans le tableau 7.2.

3) Attributs temporels (time-based features)

Ces attributs sont importants pour la détection des attaques DoS comme les *syn flooding* et les attaques Probe tels que les *scans de port*. Cette classe contient deux types d'attributs : attributs qui examinent des connexions ayant le même hôte de destination que la connexion courante durant les deux dernières secondes et attributs qui examinent des connexions ayant le même service que la connexion courante durant les deux dernières secondes. Le choix de cette fenêtre temporelle (2 secondes) est lié à la courte durée des attaques DoS et Probe car une fenêtre de 2 secondes est suffisante pour révéler les activités DoS et Probe.
Notons qu'il est possible de calculer ce type d'attributs en spécifiant d'autres fenêtres temporelles (sur 5 secondes, 10 secondes, etc.). En effet, notre outil de formatage offre la possibilité de fixer la fenêtre temporelle sur laquelle nous voulons analyser l'évolution des activités, mais celle-ci ne

Numéro	Nom de l'attribut	Description	Type
(10)	hot	nombre d'actions jugées critiques "hot"	numérique
(11)	num_failed_logins	nombre de logins échoués	numérique
(12)	logged_in	1 si le login réussit, 0 sinon	symbolique
(13)	num_compromised	nombre de conditions compromises	numérique
(14)	root_shell	1 si un shell root est obtenu, 0 sinon	symbolique
(15)	su_attempted	1 si la commande root "su" est exécutée ou tentée , 0 sinon	symbolique
(16)	num_root	nombre d'accès en mode root	numérique
(17)	num_file_creations	nombre d'opération de création de fichiers	numérique
(18)	num_shells	nombre de prompts shell	numérique
(19)	num_access_files	nombre d'opérations d'accès (écriture lecture, suppression) aux fichiers de contrôle	numérique
(20)	num_outbound_cmds	nombre de commandes non autorisées durant les sessions ftp	numérique
(21)	is_hot_login	1 si le login appartient à la liste "hot tels que root, adm, etc", 0 sinon	symbolique
(22)	is_guest_lohgin	1 si le login fait partie de la liste "guest (guest, anonymous, etc.)", 0 sinon	symbolique

TAB. 7.2: Attributs de relatifs au contenu

doit pas dépasser plusieurs secondes car si la fenêtre temporelle est trop large, il ne sera plus possible de constater les pics d'activités dus aux attaques DoS et Probe en plus du fait que cela est très coûteux en ressources de calcul et mémorisation. La liste des attributs temporels est présentée dans le tableau 7.3.

4) Attributs relatifs à un hôte particulier (host-based traffic features)

Dans [Lee 1999], l'auteur a montré que certaines attaques Probes utilisent des intervalles de temps plus longs. Ainsi, il a défini cette classe d'attributs qui sont calculés sur une plage de 100 dernières connexions par rapport à un hôte de destination particulier. Notons aussi que notre outil de formatage permet de choisir une autre plage de connexions concernant ces attributs. La liste de ces attributs est présentée dans le tableau 7.4.

5) Attributs orientés protocoles

Comme nous allons le voir dans nos études expérimentales, il est nécessaire de fournir le maximum d'attributs pertinents afin de détecter le plus grand nombre d'attaques. Ainsi, nous avons défini quelques nouveaux attributs pour révéler la présence de certaines erreurs et anomalies dans les protocoles utilisés et qui sont parfois dues à des attaques. Ces attributs sont les suivants :

1. **Bad_traffic :** cet attribut permet de révéler tout paquet qui ne respecte pas les spécifications des protocoles utilisés dans ce paquet. A titre d'exemple, cet attribut est mis à 1 à

chaque fois qu'un paquet contient des adresses non autorisées, des numéros de ports interdits, etc. Plus précisément, les événements révélés par l'attribut $Bad\_traffic$ concernent ce qui suit :

- **Taille des entêtes IP des paquets :** l'entête d'un paquet IP doit avoir une taille supérieure à 20 octets (minimal Internet header length (IHL)). De plus, la taille totale d'un paquet doit être par conséquent supérieure à la taille de l'entête IP. Si l'une de ces situations est rencontrée dans un paquet donné, cet attribut est mis à 1 pour signaler un paquet invalide ou anormal.

- **Adresses IP anormales :** Plusieurs attaques utilisent des adresses IP invalident telles que les adresses privées de classe A (10.0.0.0/0.255.255.255). Une autre anomalie concerne l'envoi de paquets TCP ou UDP comportant une adresse source IP et un numéro de port identiques à ceux de la destination tel que dans l'attaque land par exemple.

- **Ports interdits :** Comme exemple d'anomalie relative aux ports utilisés, nous citons le cas du numéro de port (source ou destination) égal à zéro. Généralement, le port source est différent du port de destination (sauf dans certains cas comme le service DNS). Si l'un de ces cas est rencontré, alors l'attribut $Bad\_traffic$ sera mis à 1.

2. **Flags incorrects :** certaines attaques sont caractérisées par l'envoi de paquets avec des combinaisons de flags incorrects comme dans l'attaque Queso). Ainsi, tout paquet n'ayant pas des combinaisons valides (telles qu'elles sont définies dans la RFC de TCP[9]) sera signalé par la mise à 1 de l'attribut Flags incorrects. Par exemple, une combinaison contenant en même temps deux parmi des flags SYN, RST et FIN est invalide et signalée par cet attribut.

3. **Direction :** un système peut être attaqué soit par des utilisateurs internes qui peuvent abuser de leur privilèges (*attaques internes*) soit par des utilisateurs externes qui tentent d'accéder d'une manière illégale à des informations ou ressources (*attaques externes*). Cet attribut nous renseigne sur la direction d'une connexion donnée par rapport au réseau surveillé. La direction d'une connexion peut être :
   - **Inbound (connexion entrante) :** toute connexion provenant d'une adresse IP ne faisant pas partie du réseau, est une connexion entrante.
   - **Outbound (connexion sortante) :** toute connexion dont l'adresse IP source fait partie du réseau surveillé mais l'adresse IP destination ne faisant partie du réseau surveillé, est une connexion sortante.
   - **Inside (connexion interne) :** toute connexion où les adresses IP source et destination appartiennent au réseau surveillé est une connexion interne.

D'autres attributs peuvent être aussi intéressants notamment pour l'administrateur réseau tels que le *timestart* (temps de début de la connexion), le nombre de paquets, les adresses IP source et destination. Mais puisque nous utilisons des modèles de classification, ce type d'attributs ne sont pas utiles pour la détection.

[9] http ://abcdrfc.free.fr/rfc-vf/rfc793.html

Numéro	Nom de l'attribut	Description	Type
(23)	count	nombre de connexions vers le même hôte que la connexion courante durant les deux dernières secondes	numérique
Les attributs suivants sont relatifs aux connexions destinés à un même hôte			
(24)	serror_rate	pourcentage de connexions ayant des erreurs "SYN"	numérique
(25)	rerror_rate	pourcentage de connexions ayant des erreurs "REJ"	numérique
(26)	same_srv_rate	pourcentage de connexions sur le même service	numérique
(27)	diff_srv_rate	pourcentage de connexions sur différents services	numérique
(28)	srv_count	nombre de connexions vers le même service que la connexion courante durant les deux dernières secondes	numérique
Les attributs suivants sont relatifs aux connexions utilisant un même service			
(29)	srv_serror_rate	pourcentage de connexions ayant des erreurs "SYN"	numérique
(30)	srv_rerror_rate	pourcentage de connexions ayant des erreurs "REJ"	numérique
(31)	srv_diff_host_rate	pourcentage de connexions sur différents hôtes	numérique

TAB. 7.3: Attributs relatifs au trafic réseau calculés sur un intervalle de deux secondes

Numéro	Nom de l'attribut	Description	Type
(32)	dst_host_count	nombre de connexions vers la même destination	numérique
(33)	dst_host_srv_count	nombre de connexions vers le même service que la connexion courante	numérique
(34)	dst_host_same_srv_rate	pourcentage de connexions sur le même service	numérique
(35)	dst_host_diff_srv_rate	pourcentage de connexions sur différents services	numérique
(36)	dst_host_same_src_port_rate	pourcentage de connexions ayant le même port source	numérique
(37)	dst_host_srv_diff_host_rate	pourcentage de connexions en provenance de différents hôtes source	numérique
(38)	dst_host_serror_rate	pourcentage de connexions ayant des erreurs "SYN"	numérique
(39)	dst_host_srv_serror_rate	pourcentage de connexions du même service et ayant des erreurs "SYN"	numérique
(40)	dst_host_rerror_rate	pourcentage de connexions ayant des erreurs "REJ"	numérique
(41)	dst_host_srv_rerror_rate	pourcentage de connexions du même service et ayant des erreurs "REJ"	numérique

TAB. 7.4: Attributs relatifs à un hôte de destination particulier pendant les 100 dernières connexions

Une fois les attributs définis, il reste à les extraire pour tester leur utilité pour la détection d'intrusions. A ce niveau, la fonction de formatage intervient afin de transformer les données brutes en données formatées décrites par les attributs définis.

7.3 Formatage du trafic réseau

Dans cette section, nous présentons notre outil de formatage de trafic réseau brut.

7.3.1 Nécessité du formatage

Les données réseaux sont de nature brute et leurs quantités sont énormes. Pour pouvoir les modéliser et les analyser avec des modèles graphiques probabilistes par exemple, il impératif de les structurer et les décrire avec des attributs pertinents et discriminants. Ainsi, les objectifs du formatage du trafic réseau se résument à ce qui suit :

- **Réduire de la quantité de données à analyser :** Réduire la quantité de données réseaux devient une tâche nécessaire du fait que les quantités de données à analyser chaque jour sont très importantes (plusieurs giga-octets dans un réseau de taille moyenne). Le trafic peut être réduit en construisant des connexions (où chaque connexion peut regrouper plusieurs paquets). A titre d'exemple, les données réseaux brutes $DARPA'98$ occupent 5 Go alors que la base $KDD'99$ qui est la version formatée de $DARPA'98$ n'occupe que 800 Mo.

- **Fournir uniquement des informations utiles et pertinentes :** Au lieu d'analyser toutes les informations disponibles dans un paquet, il est plus intéressant d'analyser uniquement les informations pertinentes et utiles.

- **Fournir suffisamment d'attributs discriminants :** il s'agit de fournir le maximum d'attributs permettant de distinguer entre activités normales et anormales. Par exemple, certains attributs tels que le protocole et le service peuvent prendre les mêmes valeurs dans des connexions normales et anormales. Il est donc impossible de distinguer entre une connexion normale et une autre anormale dans le cas où les deux connexions sont identifiées uniquement par le protocole et le service, mais leur utilisation avec d'autre attributs peut être utile.

- **Fournir des données exploitables par des techniques de l'intelligence artificielle :** Comme nous nous intéressons à appliquer des techniques de classification pour la détection d'intrusions, il est essentiel de transformer le trafic réseau du format *tcpdump* par exemple aux formats CSV qui est largement adopté par les plates-formes de fouille de données, classification, etc.

7.3.2 Description de notre outil de formatage

A notre connaissance, aucun outil de formatage n'est publiquement disponible. Nous avons alors développé notre propre outil en ajoutant de nouvelles fonctionnalités de formatage à l'analyseur de trafic réseau *Ethereal* [Orebaugh & Ramirez 2004]. Notre outil est réalisé conjointement avec K. Tabia [Benferhat *et al.* 2007] dans le cadre de l'ACI sécurité informatique DADDi[10].

[10] www.rennes.supelec.fr/DADDi/

Ethereal est un analyseur de trafic réseau qui supporte un très grand nombre de protocoles (a titre d'exemple, Ethereal 0-10-14 supporte 754 protocoles). Les principales fonctionnalités qui ont motivé le choix de Ethereal sont :

- Ethereal dispose de fonctionnalités de capture de paquets depuis plusieurs types de réseaux et des fonctionnalités de sauvegarde du trafic dans des fichiers. Il permet également d'ouvrir et d'éditer des fichiers de trafic capturé.
- Il permet de définir des filtres de capture et d'affichage.
- Il permet de reconstruire les différentes conversations entre hôtes.
- Il fournit plusieurs statistiques concernant le trafic en cours comme la répartition par protocoles, services, etc.
- Un flux applicatif TCP peut être extrait ce qui permet d'analyser uniquement les données de la couche applicative.

De plus, Ethereal est un logiciel libre et fonctionne sous la plupart des plates-formes (Unix, Linux et Windows). Il présente une interface graphique conviviale et fonctionne également en ligne de commandes (*tethereal*).

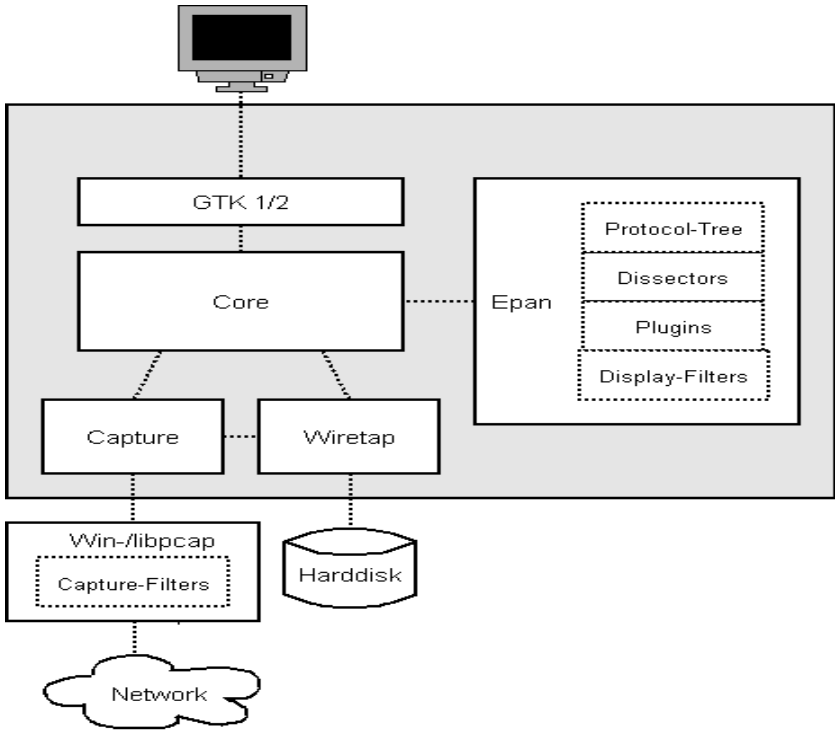

FIG. 7.2: Composants de Ethereal

La figure 7.2 montre les différents modules composant *Ethereal* et assurant ses différentes fonctionnalités. Le module *Capture* est basé sur la bibliothèque libpcap[11] (ou *winpcap* sous MS Windows), qui fournit des fonctions pour capturer les paquets réseaux. La lecture et l'enregistrement des fichiers de trafic réseau dans plusieurs formats (*tcpdump*, *pcap*, etc.) sont assurés par le module *Wiretap*. Le module *Epan (Ethereal Packet Analyzer)* regroupe plusieurs sous modules : *Display-filter* assure le filtrage des paquets réseaux affichés, les fonctions de décodage de

[11] www.tcpdump.rog

protocoles sont assurées par le module *Dessectors*, etc. Les détails des données brutes décodées et l'interface graphique sont gérés par le module *GTK*.

Comme Ethereal dispose de différentes fonctionnalités notamment pour la capture de trafic, lecture et écriture dans des fichiers ainsi que le décodage de la pile de protocoles qui composent le paquet, cela nous a aidé à implémenter nos fonctionnalités concernant le formatage de trafic réseau brut. Les nouvelles fonctionnalités de formatage que nous avons développées sont donc implémentées au niveau du module *Epan* et utilisent les fonctions de décodage de protocoles et de filtrage des paquets.

7.3.3 Fonctionnalités de notre outil

L'objectif de notre outil de formatage est de nous permettre de structurer et transformer le trafic réseau (collecté en temps réel par un sniffeur ou préalablement sauvegardé dans des fichiers) dans un format compréhensible et exploitable. Notre outil de formatage permet de construire des connexions à partir des paquets constituant ces connexions. Chaque connexion est caractérisée par les d'attributs que nous avons définis.
Nous distinguons deux types de formatage : formatage hors ligne (à partir des fichiers de données sauvegardés) et formatage en temps réel (en cours de capture). Chaque type de formatage propose deux fonctionnalités : construction de connexions finies et construction de connexions non finies.

Avant de décrire les deux types de formatage, nous décrivons d'abord la fonctionnalité de construction des connexions.

7.3.4 Construction des connexions

Selon les protocoles utilisés, trois types de connexions sont construites : TCP, UDP et ICMP. Notons que nous avons utilisé le terme connexion pour les protocoles connectés (utilisant le protocole TCP) mais aussi pour les protocoles non connectés (utilisant soit UDP soit ICMP). Nous décrivons ces connexions dans ce qui suit :

- ***Connexions TCP :*** ces connexions correspondent aux services réseaux (ssh, smtp, http, https, etc.) utilisant le protocole TCP au niveau de la couche transport. Une connexion TCP correspond à l'ensemble des paquets échangés entre une machine source et une machine destination pour réaliser un service donné. Ces connexions fonctionnent en trois phases : établissement de la connexion, transfert de données et fin de la connexion. L'établissement d'une connexion TCP se fait par une poignée de main en trois temps tandis que la fin de la connexion utilise une poignée de main en quatre temps.

  Les neuf paquets de la figure 7.3 constituent une seule connexion TCP. Comme nous l'observons sur les informations affichées dans les paquets, le service utilisé par cette connexion est http, l'établissement de la connexion est fait par l'envoi de trois drapeaux : SYN envoyé par la machine source qui initie la connexion suivi par un SYN-ACK envoyé par la machine de destination et se termine par ACK pour accuser la réception de l'établissement de la connexion.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	194.7.248.153	172.16.114.50	TCP	25104 > http [SYN] Seq=0 Ack=0 Win=512 Len=0 MSS=1460
2	0.001049	172.16.114.50	194.7.248.153	TCP	http > 25104 [SYN, ACK] Seq=0 Ack=1 Win=31744 Len=0 MSS=1460
3	0.001213	194.7.248.153	172.16.114.50	TCP	25104 > http [ACK] Seq=1 Ack=1 Win=32120 Len=0
4	0.001772	194.7.248.153	172.16.114.50	HTTP	GET /people/yoav/backgrounds/blue_rock.gif HTTP/1.0
5	0.005369	172.16.114.50	194.7.248.153	HTTP	HTTP/1.0 404 Not found (text/html)
6	0.005487	172.16.114.50	194.7.248.153	TCP	http > 25104 [FIN, ACK] Seq=282 Ack=335 Win=31744 Len=0
7	0.005569	194.7.248.153	172.16.114.50	TCP	25104 > http [ACK] Seq=335 Ack=283 Win=31838 Len=0
8	55.353320	194.7.248.153	172.16.114.50	TCP	25104 > http [FIN, ACK] Seq=335 Ack=283 Win=32120 Len=0
9	55.354167	172.16.114.50	194.7.248.153	TCP	http > 25104 [ACK] Seq=283 Ack=336 Win=31744 Len=0

FIG. 7.3: Exemple de paquets constituant une connexion TCP

- ***Connexions UDP :*** le protocole UDP se trouve au même niveau que le protocole TCP (couche transport), mais les services qui l'utilisent pour transporter les data-grammes fonctionnent en mode non connecté. Une connexion UDP correspond à l'ensemble des paquets générés lors du transfert des données entre deux entités, chacune étant définie par une adresse IP et un numéro de port. La figure 7.4 montre une liste de paquets UDP générée lors d'une requête/réponse DNS.

Source	Destination	Protocol	Info
192.168.1.30	192.168.0.40	DNS	Standard query ANY pascal.victim
192.168.0.40	192.168.1.30	DNS	Standard query response A 192.168.0.20

FIG. 7.4: Exemple de paquets constituant une connexion UDP

Après le formatage, ces deux paquets seront représentés par une seule connexion de type UDP. Le premier paquet représente la requête qui est envoyée par la machine source et le deuxième représente la réponse à la requête.

- ***Connexions ICMP :*** le protocole ICMP fonctionne en mode non connecté permettant par exemple de fournir des messages de contrôle de flux et de signaler des erreurs d'acheminement de paquets. Une connexion ICMP peut être une requête/réponse ICMP concernant par exemple un test de connectivité *ping* et sa réponse (voir figure 7.5) ou un seul paquet dans le cas ou ICMP est utilisé pour signaler une erreur.

Source	Destination	Protocol	Info
192.168.1.30	192.168.0.20	ICMP	Echo (ping) request
192.168.0.20	192.168.1.30	ICMP	Echo (ping) reply

FIG. 7.5: Exemple de paquets constituant une connexion ICMP

7.3.5 Types de formatage

Notre outil offre deux possibilités de formatage du trafic réseau brut :

1. **Formatage hors ligne :** ce type de formatage consiste à construire des connexions finies (complètes ou terminées) ou non finies (incomplètes ou pas encore terminées) à partir du trafic réseau capturé et enregistré dans des fichiers. Le formatage hors ligne est intéressant et utile dans le cadre de la détection d'intrusions dans la mesure où la construction des connexions permet d'analyser et de retracer tous les événements d'une connexion, ce qui permet une meilleure analyse que lorsqu'on considère des paquets séparément. En plus, la détection en temps réel est confrontée à l'analyse de grande quantité de données, ce qui dans la pratique exige une analyse rapide au détriment par exemple de l'exhaustivité. L'analyse hors ligne est complémentaire et permet d'analyser plus profondément et systématiquement car la rapidité n'est pas une contrainte ici.

2. **Formatage en temps réel :** afin de réaliser une détection en temps réel, il est nécessaire de formater le trafic brut en temps réel également. Comme le formatage hors ligne, le formatage en temps réel permet de construire des connexions finies ou non finies mais à partir du trafic en cours de capture. La fonctionnalité de formatage en temps réel est indispensable pour l'analyse en temps réel, notamment pour prendre des contres-mesures le plus tôt possible.

Une connexion finie est caractérisée par les attributs calculés sur l'ensemble des paquets appartenant à cette connexion. Pour construire des connexions finies, notre outil crée une nouvelle connexion dès la réception de son premier paquet, et la met à jour à chaque fois qu'un nouveau paquet appartenant à cette connexion est reçu. Nous donnons dans ce qui suit un exemple concernant la construction d'une connexion finie de type TCP.

Exemple 7.1 (Connexion finie de type TCP)
Notre outil permet de construire à partir de la liste de paquets présentée dans la figure 7.3 la connexion correspondante et de la décrire par un ensemble d'attributs. A titre exemple, nous avons dans les trois premiers paquets des échanges (SYN, SYN-ACK et ACK), cela signifie que la connexion est établie d'une façon normale et dans les quatre derniers paquets, nous avons les flags (FIN-ACK, ACK, FIN-ACK et ACK), ce qui signifie également la terminaison normale de la connexion. Donc, l'état de la connexion est représenté par l'attribut flag qui prend la valeur "SF". A partir des informations dans les paquets, le service utilisé par cette connexion est "http", les adresses IP source et destination sont "194.7.248.153" et "172.16.114.50" , etc. La connexion finie construite est donc telle qu'elle est affichée dans le tableau suivant (nous donnons quelques attributs uniquement) :

Durée	Protocole	Service	Flag	Src-bytes	Dst-bytes	land	Direction	...
55.354	tcp	http	SF	334	281	0	inbound	...

Tab. 7.5: Exemple d'une connexion TCP finie

Une connexion non finie est une connexion qui concerne seulement un ensemble de paquets à un certain instant (connexion qui n'a pas encore atteint sa fin). Les attributs qui la décrivent sont les mêmes que ceux d'une connexion finie sauf que certains attributs prennent des valeurs décrivant la connexion à cet instant. Comme exemples d'attributs évoluant dans le temps, on trouve : durée, flag, etc. Concernant la construction des connexions non finies, notre outil offre deux possibilités :

- La première possibilité de construire des connexions non finies concerne la mise à jour d'une connexion à chaque fois qu'un nouveau paquet appartenant à cette connexion est capturé. L'arrivé d'un nouveau paquet peut apporter de nouvelles informations à une connexion. Dans l'exemple de la figure 7.3, dès la réception du premier paquet, notre outil crée une nouvelle connexion dont les attributs la décrivant sont (durée = 0 secondes, Protocole est TCP, Service est http, Flag = S0, etc), l'arrivée du deuxième paquet apporte de nouvelles informations et change l'état de cette connexion, c'est-à-dire, certains attributs la décrivant prennent de nouvelles valeurs tel que (durée = 0.001 secondes, Protocole est TCP, Service est http, Flag = S0, etc), le nouvel état de la connexion à l'arrivé du quatrième paquet est (durée = 0.0017 secondes, Protocole est TCP, Service est http, Flag = S1, etc). Ainsi, à ce niveau, la connexions n'est pas terminée (seulement quatre paquets), elle est donc considérée incomplète.

- La deuxième façon de construire des connexions non finies concerne l'utilisation d'un paramètre temporel afin de générer les connexions chaque T secondes. Voir l'exemple 7.2.

Exemple 7.2 (Connexion incomplète de type TCP)
Cet exemple illustre l'état de la même connexion générée à différents instants (T=1 seconde, T=2 seondes, T=4 secondes, etc.).

T (secondes) :	Durée	Protocole	Service	Flag	Src-bytes	Dst-bytes	land	hot	...
T=1 :	**0.986**	tcp	telnet	**S1**	**87**	**51**	0	**0**	...
T=2 :	**1.614**	tcp	telnet	**S1**	**93**	**116**	0	**0**	...
T=4 :	**3.824**	tcp	telnet	**S1**	**108**	**135**	0	**1**	...
...	...	...	...	...	...	...	...	...	...
T=15 :	**14.804**	tcp	telnet	**S1**	**123**	**202**	0	**2**	...
...	...	...	...	...	...	...	...	...	...
T=30 :	**865**	tcp	telnet	**SF**	**139**	**249**	0	**3**	...

TAB. 7.6: Exemple montrant l'évolution d'une connexion en fonction du temps

7.4 Données d'expérimentations

Notre objectif dans le reste de ce chapitre est, d'une part, utiliser un classifieur bayésien afin de montrer l'utilité de la fonction de formatage et d'extraction d'attributs et, d'autre part,

montrer qu'il est possible de détecter certaines catégories d'intrusions avec des connexions incomplètes. Nos études expérimentales sont réalisées sur une partie de la base de données $DARPA'99$ [Darpa 1999]. Étant donné que ces données sont brutes, nous avons donc utilisé l'outil de formatage présenté dans ce chapitre pour les formater en connexions. Les données que nous avons utilisées et les résultats expérimentaux sont présentés dans les sections suivantes.

7.4.1 Description des données DARPA'99

L'agence américaine DARPA (Defence Advanced Research Projects Agency) a lancé un programme afin d'élaborer des méthodologies d'évaluation pour les systèmes de détection d'intrusions. La première campagne d'évaluation dans ce programme a eu lieu en 1998 et réalisée par les laboratoires MIT Lincoln (Massachusetts Institute of Technology). La deuxième campagne a eu lieu en 1999 et une troisième en 2000. L'objectif de ces campagnes était de fournir une base d'évaluation contenant du trafic normal et différentes variantes d'attaques que les chercheurs travaillant sur la détection d'intrusions peuvent utiliser. Lors de la conférence mondiale sur la découverte de connaissances dans les bases de données KDD 1999, une compétition concernant la détection d'intrusions a eu lieu [Elkan 2000]. Pour les besoins de la compétition, l'un des participants [Lee *et al.* 1999] a fourni la base de données KDD'99 [KDD 1999], qui est une version formatée des données brutes DARPA'98. Depuis, la base de données KDD'99 a été largement utilisée en détection d'intrusions, notamment lorsque des techniques de fouille de données sont utilisées. Contrairement aux données DARPA'98 pour lesquelles une base de données formatée est disponible, aucune base formatée n'est disponible pour DARPA'99. C'est pourquoi, ces données n'ont pas connu une large utilisation comme DARPA'98 bien qu'elles contiennent plusieurs variantes d'attaques, notamment les nouvelles. Concernant les données DARPA'2000, elles sont très utilisées en corrélation d'alertes [Yu & Frincke 2007, Ning *et al.* 2004].

Nous avons choisi de travailler avec les données DARPA'99 car, d'une part, elles contiennent un grand nombre de nouvelles attaques, et d'autre part, les données KDD'99 contiennent quelques incohérences [Bouzida 2006]. A titre d'exemple, il existe dans la base de test KDD'99 des connexions complètement identiques (c-à-d. que les attributs les caractérisant sont les mêmes), mais certaines sont des connexions normales, alors que les autres représentent l'attaque *snmpget*. Ce type d'incohérences pose un problème sérieux pour la qualité de la détection.

Les données DARPA'99 [Darpa 1999] comportent cinq semaines de trafic réseau et autres données d'audit fournies par les laboratoires Lincoln. Les données des trois premières semaines sont réservées pour l'apprentissage et celles des deux dernières semaines sont fournies pour le test. Les cinq semaines sont décrites comme suit :

- Les données collectées pendant les deux semaines 1 et 3 contiennent uniquement du trafic normal. Toutes ces données sont réservées pour l'apprentissage.
- Les données de la semaine 2 contiennent du trafic normal et différents types d'attaques. Elles sont également destinées pour l'apprentissage.
- Les deux dernières semaines (4 et 5) contiennent des données de test contenant du trafic normal et plusieurs attaques présentes également dans la base d'apprentissage, mais également plusieurs nouvelles attaques (qui ne sont pas présentes dans la base d'apprentissage).

Les traitements que nous avons effectués sur les données brutes sont les suivants :

– **Formatage des données :** nous avons utilisé notre outil de formatage pour construire des connexions à partir des données brutes. Deux types de connexions sont générés : connexions finies et connexions non finies. Toutes les connexions sont décrites par les attributs que nous avons décrits. Les statistiques concernant une partie des données formatées sont résumées dans le tableau 7.7.

	Données d'apprentissage		Données de test	
	Effectif	%	Effectif	%
Normal	455479	91.16%	456241	85.40%
DoS	21763	4.36%	66436	12.44%
R2L	1890	0.38%	1060	0.20%
U2R	793	0.16%	52	0.01%
Probe	19725	3.95%	10425	1.95 %
Data	2	0.0004%	3	0.0004%
Total	**499652**	**100%**	**534217**	**100%**

TAB. 7.7: Distributions des catégories de connexions dans le trafic formaté de Darpa'99

– **Étiquetage des connexions :** l'étiquetage des connexions consiste à attribuer à une connexion anormale le *nom de l'attaque* (smurf, yaga, neptune, etc.) ou sa catégorie d'attaque (DoS, Probe, R2L, U2R ou Data) ou bien *Normal* à une connexion normale. Cette opération est fastidieuse à cause du nombre de connexions à étiqueter manuellement et à cause des informations insuffisantes fournies dans DARA'99[12]. Un exemple de ces informations est présenté dans le tableau 7.8.

ID : 41.182453
Date : 03/29/1999
Name : New_Attack_1999 (secret)
Category : data
Start_Time : 18 :24 :19
Duration : 00 :08 :41
Attacker : 195.115.218.108
Victim : 172.016.112.050
Username : abramh
Ports :
At_Attacker : 25{1}
At_Victim : 23{1}, 113{1}

TAB. 7.8: Exemple d'informations décrivant les données de test DARPA'99

Les informations représentées dans le tableau 7.8 indiquent que dans la journée du 29/03/1999 de la semaine 4 (données de test), à 18 :24 :19 (timestart), trois connexions ont été générées par l'attaque *secret*. L'adresse IP source de la première est "172.16.112.50",

[12]http ://www.ll.mit.edu/IST/ideval/docs/docs index.html

l'adresse IP destination est "195.115.218.108", le service est smtp (25). Les deux autres connexions sont de type *telnet (23), ident (113)* et les adresses IP source (resp. destination) sont "195.115.218.108" (resp. "172.16.112.50").
Après formatage, nous avons retrouvé ces connexions, et nous avons étiqueté chacune par le nom de l'attaque "*secret*". Cette attaque est nouvelle et appartient à la catégorie *"Data"*.

La base de données DARPA'99 contient plus de 200 instances de 58 attaques différentes appartenant à l'une des catégories suivantes :

1. **Les dénis de service (Denial of Service (DoS))** : dans cette catégorie d'attaques, le but de l'attaquant est d'empêcher le fonctionnement normal du réseau ou d'un service en saturant les ressources. Un DoS peut avoir pour cible un serveur, une machine, etc.
2. **Les attaques U2R (User to Root)** : cette catégorie d'attaques concerne un abus de privilèges de la part d'un utilisateur autorisé pour accéder aux privilèges du super utilisateur.
3. **Les attaques R2L (Remote to User)** : des attaquants n'ayant pas un accès autorisé sur la machine cible, arrivent à accéder aux ressources du système comme s'ils étaient des utilisateurs autorisés.
4. **Les Probes** : les attaquants procèdent à une phase de collecte d'informations recherchant des cibles potentielles. La collecte d'informations peut s'effectuer de plusieurs manières comme lors d'un scan de ports.
5. **Data** : cette catégorie d'attaques concerne le transfert de fichiers dans des locations illégales et non conformes à la politique de sécurité adoptée. Il existe très peu d'attaques Data dans les bases de test et d'apprentissage de DARPA'99.

   Les attaques des données DARPA'99 sont présentées dans le tableau 7.9 :

Catégorie de l'attaque	Nom de l'attaque
Probe	ipsweep, mscan, NTinfoscan, nmap, queso, portsweep, saint, satan, resetscan
DoS	apache2, arppoison, back, Crashiis, land, mailbomb, neptune (SYN Flood), pod (Ping of Death), processtable, selfping, smurf, sshprocesstable, Syslogd, tcpreset, teardrop, udpstorm
R2L	dictionary, ftpwrite, guest, httptunnel, imap, named ncftp, netbus, netcat, phf, ppmacro, sendmail, sshtrojan, xlock, Xsnoop
U2R	anypw, casesen, eject, ffbconfig, fdformat, loadmodule, ntfsdos, perl, ps, sechole, xterm, yaga
Data	secret, ntfsdos, ppmacro

TAB. 7.9: Répartition des attaques dans DARPA'99

7.5 Résultats d'expérimentations sur les données DARPA'99

Dans cette section, nous présentons deux types d'expérimentations : détection avec des connexions finies et détection avec des connexions non finies. Dans la première expérimentation, nous considérons une base de test et une base d'apprentissage où chacune contient des connexions finies. L'objectif de cette expérimentation est de tester l'importance des attributs que nous avons définis (les 41 attributs de type KDD'99 et les attributs orientés protocoles). Dans la seconde expérimentation, nous considérons uniquement les connexions qui sont bien détectées dans la première expérimentation mais générées à des instants différents (à T= 0.5, T=2 secondes, etc.) pour voir s'il est possible de les détecter en temps réel (avec des connexions non finies). Nous utilisons donc plusieurs bases de test contenant les mêmes connexions sauf que dans chaque base les connexions sont générées à des instants différents. Nous donnons dans ce qui suit les paramètres et critères d'évaluation que nous utiliserons pour l'évaluation de nos résultats :

- Le taux de classification est le pourcentage des connexions correctement classifiées (PCC) par rapport au nombre total de connexions. Le PCC est calculé comme suit :

$$PCC = \frac{Nombre \quad de \quad connexions \quad correctement \quad \text{classifiées}}{Nombre \quad total \quad de \quad connexions} * 100 \tag{7.1}$$

- Le taux de vrais positifs (TP) est calculé comme suit :

$$TP = \frac{Nombre \quad d'attaques \quad correctement \quad \text{détectées}}{Nombre \quad total \quad d'attaques} * 100 \tag{7.2}$$

- Le taux de fausses alertes ou de faux positifs (FP) est calculé comme suit :

$$FP = \frac{Nombre \quad de \quad fausses \quad alertes}{Nombre \quad total \quad de \quad connexions \quad normales} * 100 \tag{7.3}$$

7.5.1 Détection d'intrusions avec des connexions complètes

Pour ces expérimentations, nous utilisons le classifieur bayésien naïf (RBN) décrit au chapitre 6. Nous avons réalisé deux expérimentations avec uniquement des connexions finies. Dans la première, les connexions sont décrites par les 41 attributs KDD'99. Pour la seconde expérimentation, les connexions sont décrites par 44 attributs (les 41 attributs de type KDD'99 plus les trois nouveaux attributs). Les résultats concernant la première expérimentation sont donnés dans le tableau 7.10.

	Taux de détection (PCC)					
	Normal	DoS	Probe	R2L	U2L	Data
Normal	**99.30%**	0.24%	0.40%	0.02%	0.04%	0.00% %
DoS	41.70%	**48.40%**	9.89 %	0.01%	0.00%	0.00%
Probe	4.01%	0.30%	**95.50%**	0.07%	0.12%	0.00%
R2L	83.20%	0.00%	0.00	**16.80%**	0.00%	0.00%
U2R	71.42%	0.00%	0.00%	3.58 %	**25.00%**	0.00%
Data	66.66%	0.00%	0.00%	0.00%	0.00%	**33.33%**
PCC total	**92.21%**					
% Vrais positifs (TP)	**70.44%**					
% Faux positifs (FP)	**0.70%**					

TAB. 7.10: Résultats du RBN dans la classification des connexions finies décrites avec 41 attributs

Les résultats de la deuxième expérimentation (où les connexions sont décrites par 44 attributs) sont présentés dans le tableau 7.11.

	Taux de détection (PCC)					
	Normal	DoS	Probe	R2L	U2L	Data
Normal	**99.30%**	0.23%	0.39%	0.62%	0.06%	0.00% %
DoS	41.35%	**52.90%**	5.74 %	0.01%	0.00%	0.00%
Probe	1.15%	0.25%	**98.26%**	0.12%	0.12%	0.00%
R2L	60.73%	0.00%	0.20	**39.27%**	0.00%	0.00%
U2R	69.04%	0.00%	0.00%	5.95 %	**25.00%**	0.00%
Data	66.66%	0.00%	0.00%	0.00%	0.00%	**33.33%**
PCC total	**93.01%**					
% Vrais positifs (TP)	**71.81%**					
% Faux positifs (FP)	**0.70%**					

TAB. 7.11: Résultats du RBN dans la classification des connexions finies décrites avec 44 attributs

Discussions

Concernant les résultats de détection avec 41 attributs, nous remarquons de faibles taux de détection au niveau des catégories d'attaques DoS (48.40%), R2L (16.80%), U2R (25%) et Data (33.33%). Par contre, le taux de classification des attaques Probe est assez élevé. Concernant les

attaques R2L, U2R et Data, ces résultats s'expliquent par la distribution des attaques dans la base de test dont la majorité sont nouvelles ou différentes des attaques de la base d'apprentissage. A titre d'exemple, la base de test contient plus de 80% d'attaques R2L qui sont nouvelles et complètement différentes des attaques R2L d'apprentissage. D'autre part, les catégories d'attaques R2L et U2R se caractérisent par des proportions très faibles dans la base d'apprentissage (uniquement 0.20% d'attaques R2L et 0.01% d'attaques U2R). Quant aux attaques Data, elles sont nouvelles et ne représentent que 0.0004% dans les bases de test et d'apprentissage.

Pour ce qui est des attaques DoS, elles ne sont pas bien détectées même si la majorité de ces attaques sont connues (les nouvelles attaques représentent environs 0.87%). Cela est dû aux caractéristiques des attributs de ces attaques qui sont différentes dans les deux bases. Nous considérons le cas des attaques *Neptune* qui représentent une grande partie des attaques DoS pour illustrer pourquoi ces attaques ne sont pas bien classifiées. Dans la base de test, l'attribut "*Flag*" de cette attaque prend la valeur "S0", alors que dans la base d'apprentissage, cet attribut prend la valeur "REJ". Ainsi, lors de la phase d'apprentissage, la probabilité de l'attribut "Flag = S0" dans le contexte de la classe DoS est nulle. Donc, toutes les connexions *Neptune* ne seront pas correctement classifiées dans la base de test car la probabilité a posteriori de la catégorie DoS étant donné le flag S0 est nulle.

Concernant la comparaison des résultats présentés dans les tableaux 7.10 et 7.11, nous observons que le taux de détection s'est relativement amélioré au niveau de certaines catégories d'attaques lorsque les connexions sont décrites par 44 attributs. Nous remarquons qu'au niveau de la classe normale, le taux de classification est le même dans les deux cas, ce qui signifie que les nouveaux attributs n'apportent pas d'amélioration pour les connexions normales. Au niveau des attaques DoS, le taux de détection s'est amélioré de 48.4% à 52.9%. Ce taux n'est pas négligeable car nous avons environs 800 connexions DoS qui ne sont pas bien détectées lorsque nous utilisons uniquement 41 attributs, alors qu'elles sont correctement détectées lorsque nous considérons les nouveaux attributs. En effet, l'attribut direction contribue à la détection des attaques DoS.
Concernant les attaques Probe, nous constatons également une amélioration du taux de détection de 95.50% à 98.26%. Les attributs *Bad-traffic* et *Flags-incorrects* influencent le taux de détection des attaques Probe où nous constatons une amélioration significative malgré le petit nombre de connexions qui comportent par exemple des flags TCP incorrects (uniquement quelques attaques Queso). Le taux de détection des attaques R2L s'est amélioré en utilisant les 44 attributs mais ce taux est assez faible. Cela est dû aussi au nombre négligeable de ces attaques par rapport aux connexions normales, DoS et Probe. Concernant les attaques U2R et Data, le taux de détection ne s'est pas amélioré. Le taux de faux négatifs est assez élevé dans les deux expérimentations car la majorité des attaques non détectées sont classées comme des connexions normales.

Les résultats des tableaux 7.10 et 7.11 permettent de dire que les nouveaux attributs apportent une amélioration concernant le taux de détection. De plus, le taux de faux positifs n'a pas augmenté dans le cas des 44 attributs (il est égal à 0.70% dans les deux expérimentations). Même si les attributs que nous avons définis permettent de détecter plusieurs types d'attaques, le taux de faux positifs ou fausses alertes reste cependant important (0.70%). C'est pourquoi, différentes approches de corrélation d'alertes ont été proposées afin de remédier à ce problème. Pour notre part, nous avons proposé une nouvelle approche pour la corrélation d'alertes. Nous la présenterons dans le chapitre suivant.

7.5.2 Détection d'intrusions avec des connexions incomplètes

Tous les travaux cités dans le chapitre précédent et d'autres traitant de la détection d'intrusions ont travaillé sur des connexions finies. Les résultats sont généralement satisfaisants, mais la détection est toujours réalisée à la fin des connexions, c'est-à-dire que les données sont considérées complètes et certaines.

L'objectif de travailler avec des connexions non finies est de détecter les intrusions en temps réel ou anticiper la détection. Prenons l'exemple d'une attaque buffer-overflow contre un serveur Web. Cette attaque vise à provoquer un débordement de tampon dans le but d'écraser des parties du code d'une application ou d'injecter des commandes ou des shell codes. Si par exemple, nous constatons dès les premiers paquets, que la taille des données transférées est très grande par rapport à la taille habituelle d'une requête http normale, il faudra alors réagir rapidement pour prendre les contres-mesures nécessaires contre une telle activité. Ainsi, pour détecter des attaques en temps réel, il est nécessaire par exemple de formater le trafic réseau en temps réel. Pour ce faire, il faut construire des modèles appropriés pour analyser les connexions en temps réel. Pour cela, nous avons réalisé plusieurs expérimentations sur la même base de test, mais les connexions sont générées à des durées différentes. Les caractéristiques des données et expérimentations réalisées sont comme suit :

- La première étape consiste à sélectionner seulement des connexions finies bien détectées dans la première expérimentation (détection avec des connexions complètes dans le cas où les connexions sont décrites par 44 attributs). Nous avons pris 99.72% des connexions correctement classifiées dans le but de voir si ces connexions seront aussi bien détectées avant leur fin. Nous avons sélectionné seulement des connexions ayant une durée supérieure à 0 secondes. Toutes les connexions icmp (qui contiennent généralement un seul paquet) par exemple ne sont pas utilisées, car pour ces connexions, le problème de détection en temps réel ne se pose pas. Soit $Base_{finie}$ la base qui contient toutes les connexions sélectionnées dans cette première étape.

- Formater les données de test selon différentes durées spécifées (à T=0.5 secondes, T=2, T=4, T=10, T=25, T=50 et T= 100 secondes).

- Soient $Base_{T=0.5}$ (rep. $Base_{T=2}$, $Base_{T=4}$, $Base_{T=10}$, $Base_{T=25}$, $Base_{T=50}$ et $Base_{T=100}$) des bases qui représentent les mêmes connexions que celles dans $Base_{finie}$, sauf que les connexions de ces bases sont non finies, c'est-à-dire générées à T=0.5 secondes (resp. T=2, T=4, T=10, T=25, T=50, T= 100).

- Étiqueter chaque connexion (normale ou attaque) comme nous l'avons déjà présenté dans la section 7.4.1.

Notons aussi que l'évaluation de nos résultats est basée sur la comparaison des résultats obtenus en utilisant un réseau bayésien naïf (RBN) sur le même ensemble de connexions lorsqu'elles sont finies et lorsque ces connexions sont non finies.

Les tableaux 7.12 et 7.13 présentent les résultats de la classification sur le même ensemble de connexions (le tableau 7.12 présente les résultats des connexions finies et le tableau 7.13 présente les résultats des connexions non finies).

→	Normal	DoS	Probe	U2R	R2L	Data
PCC	99.80 %	99.79%	100 %	100%	96.09 %	100%
PCC Total	**99.72%**					

TAB. 7.12: Résultats de classification des connexions finies en utilisant un RBN

.

→	Normal	DoS	Probe	U2R	R2L	Data
T=0.5 secondes	97.83%	98.27%	100%	77.27%	98.55%	0.00%
PCC Total	**97.85%**					
T=2 secondes	97.81%	98.79%	100%	95.45%	98.84%	100.00%
PCC Total	**97.88 %**					
T=4 secondes	97.48%	99.15%	100%	90.90%	99.13 %	0.00%
PCC Total	**97.60%**					
T=10 secondes	97.98%	99.07%	100%	95.45%	98.98%	100.00%
PCC Total	**97.95%**					
T=25 secondes	98.38%	99.11%	100%	95.45%	95.94%	100.00 %
PCC Total	**98.37%**					
T=50 secondes	98.35%	99.15%	100%	90.90%	96.09%	100.00%
PCC Total	**98.34%**					
T=100 secondes	98.46%	99.27%	100%	97.18%	95.65%	100.00%
PCC Total	**98.45%**					

TAB. 7.13: Résultats de classification des connexions non finies en utilisant un RBN

Le tableau 7.14 présente les résultats de la classification de toutes les connexions normales classifiées normales et les connexions anormales classifiées anormales.

Le tableau 7.15 présente les résultats de détection de certaines connexions extraites des bases de données utilisées. Notons que le signe (-) signifie que les connexions sont finies.

T (durées spécifiées)	Normal	Attaque
T= 0.5 secondes	97.83 %	98.49 %
T= 2 secondes	97.81 %	98.95 %
T= 4 secondes	97.48 %	99.14 %
T= 10 secondes	97.98 %	99.14 %
T= 25 secondes	98.38 %	99.14 %
T= 50 secondes	98.35 %	99.17 %
T= 100 secondes	98.45 %	99.29 %
T= fin des connexions	99.80%	99.75 %

TAB. 7.14: Pourcentage de la classification des connexions normales et attaques

7.5.3 Analyse des résultats obtenus

Les résultats des tableaux 7.12, 7.13, 7.14 et 7.15 montrent que la classification des connexions non finies donne des résultats satisfaisants. En particulier, dans les catégories d'attaques DoS, R2L, U2R et Probe, le taux de détection est assez élevé dans les différents cas. On remarque aussi que le taux de détection des attaques R2L est meilleur dans les cas des connexions incomplètes (98.55% à T=0.5 secondes) que lorsque les connexions sont finies (96.09%). Pour expliquer ce fait, considérons l'exemple suivant : notre base contient 39 connexions guesstelnet (19 ayant le flag SF et 20 ayant le flag RST0 et la durée moyenne de ces connexions est supérieure à 10 secondes lorsqu'elles sont finies). Nous constatons que lorsque ces connexions sont finies, 19 instances sont correctement classifiées et les 20 autres sont mal classifiées. Cependant, lorsque les les 39 connexions sont non finies (T = 0.5, 2, 4 et 10 secondes), elles sont toutes correctement classifiées. A titre d'exemple, lorsque T=0.5 secondes, le flag des 39 connexions guesstelnet est $S1$ et la durée de chacune de ces connexions est inférieure à 0.5 secondes. Cependant, lorsqu'elles sont finies, le flag des 20 instances incorrectement classifiées est SF et la durée de chacune d'elles est supérieure à 10 secondes. Le flag des instances correctement classifiées (19 instances) est $RSTO$ et la durée est supérieure à 10 secondes. Les autres résultats s'expliquent de la même manière pour T=25, T= 50 et T=100 secondes (à ces moments, les 39 connexions guestelnet sont déjà finies) où on observe aussi une légère baisse du taux de détection des attaques R2L (voir Tableau 7.13). En effet, quand T=0.5 secondes, toutes les connexions ont les mêmes caractéristiques, c'est pour cela qu'elles sont correctement classifiées, et lorsqu'elles sont finies, les 19 instances incorrectement classifées ressemblent à une autre attaque.

Concernant les connexions normales, on remarque que lorsque les connexions sont finies, le taux de classification est plus élevé que lorsque les connexions sont incomplètes. Par exemple, à T =0.5 secondes, 2.17% des connexions normales sont classifiées comme attaques. Ce résultat est dû au fait que beaucoup de connexions anormales lorsqu'elles démarrent (premiers paquets) sont similaires aux connexions normales (car les signatures d'attaques ne sont pas encore observées). Comme nous le constatons dans le tableau 7.15, à T=0.5 secondes, seulement 2 connexions back sont correctement classifiées à ce moment là (la signature de l'attaque back n'est pas encore observée). D'après le tableau 7.15, nous pouvons constater que certaines attaques sont détectées

Nom de l'attaque	# d'instances	Catégorie	Durée	T= 0.5	T= 2	T= 4	T= 10	T= 25	T= 50	T= 100	T= Fin
back	13	DoS	> 200 s	2	4	10	10	11	12	12	-
mailbomb	978	DoS	> 1.10 s	964	972	-	-	-	-	-	-
apache2	1315	DoS	> 900 s	1299	1298	1298	1295	1297	1295	1296	-
secret	1	Data	= 492.225 s	1	1	0	1	1	1	1	-
satan	43	Probe	> 3 s	43	43	-	-	-	-	-	-
sqlattack	2	U2R	> 68 s	1	2	2	2	2	2	-	-
guesstelnet	39	R2L	> 10 s	39	39	39	39	20	20	20	20
guessftp	73	R2L	>1.50 s	73	-	-	-	-	-	-	-
netbus	3	R2L	> 5 s	0	3	3	-	-	-	-	-
xlock	4	R2L	> 13 s	2	4	4	4	4	-	-	-
guesspop	30	R2L	> 4 s	30	30	30	-	-	-	-	-
ffbconfig	1	U2R	= 24699.352 s	1	1	1	1	1	1	1	-
ffbconfig	1	U2R	= 478 s	0	1	1	1	1	1	0	-
selfping	1	DoS	= 2.79 s	1	1	1	1	1	1	1	-
xerm	2	U2R	> 36 s	1	2	2	2	2	2	1	-
eject	1	U2R	= 27.831 s	1	1	1	1	1	-	-	-
eject	1	U2R	= 992.743 s	0	0	0	0	0	0	1	-
sshtrojan	1	R2L	= 628.664 s	0	1	1	1	1	1	1	-
perl	1	U2R	= 643.257 s	1	1	1	1	1	1	1	-
httptunnel	4	R2L	> 7 s	2	2	2	-	-	-	-	-
processtable	175	DoS	> 100 s	175	175	175	175	175	175	175	-
snpmget	500	R2L	> 4 s	500	500	500	-	-	-	-	-

TAB. 7.15: Résultats de détection de quelques attaques non finies

juste après les premiers paquets (satan, snmpget, processtable) tandis que d'autres demandent plus de temps (back, eject).

7.6 Conclusion

Dans ce chapitre, nous avons abordé deux éléments essentiels pour le problème de détection d'intrusions : définition d'attributs et formatage du trafic réseau brut. La définition d'attributs

est une étape importante pour pouvoir identifier la nature des événements ayant lieu dans un réseau. De plus, comme notre travail consiste à modéliser le problème de détection d'intrusions avec des modèles graphiques probabilistes, cette étape est indispensable afin de pouvoir construire nos modèles.

Nous avons présenté également des résultats expérimentaux où, d'une part, nous avons testé l'utilité de notre outil de formatage pour transformer les données brutes en données formatées et également l'utilité des attributs que nous avons définis. D'autre part, nous avons fourni quelques résultats préliminaires concernant la détection avec des connexions incomplètes. L'idée principale était de montrer qu'il est possible de détecter le plus tôt possible des connexions suspectes au sein d'un réseau (avant la fin des connexions). Les résultats que nous avons obtenus sont prometteurs, notamment pour plusieurs types d'attaques. Cependant, comme nous l'avons signalé dans les expérimentations concernant la détection avec des connexions finies, le taux de fausses alertes reste important. C'est pourquoi, nous avons proposé une approche de corrélation d'alertes permettant de remédier à ce problème. Nous présentons cette approche dans le chapitre suivant.

Chapitre 8

Application de la logique $\mathcal{MQCL}$ à la corrélation d'alertes

Sommaire

8.1 Introduction

Malgré l'efficacité des différentes approches de corrélation d'alertes développées pour éliminer les informations redondantes ou détecter des attaques coordonnées et complexes, elles continuent à produire un grand nombre d'alertes. A notre connaissance, aucune des approches de corrélation d'alertes existantes ne tient en compte les préférences et les connaissances d'un administrateur réseau. L'approche que nous proposons suit une autre direction par rapport aux approches existantes, mais elle est complémentaire dans le sens où elle permet également de réduire le nombre d'alertes générées dans le but de faciliter la tâche de l'administrateur qui surveille le système. Notre approche consiste à modéliser les connaissances que l'administrateur a sur le système, le réseau, l'IDS, les alertes et ses préférences concernant le type d'alertes qu'il préfère analyser ou ignorer. L'idée principale, est de développer une logique qui permet d'ordonner les alertes et de les classer dans des catégories différentes selon les connaissances et les préférences de l'administrateur. Les alertes qui seront classées dans la première catégorie sont celles qui devraient satisfaire les connaissances et les préférences de l'administrateur. Si nécessaire, des alertes de la deuxième catégorie seront présentées, et ainsi de suite.

La logique que nous avons utilisée pour développer notre approche concerne l'extension en fragment de la logique du premier ordre du langage des logiques que nous avons présentées dans la première partie de cette thèse (chapitres 3 et 4). Ainsi, selon le besoin de l'application, il est suffisant d'appliquer la logique $\mathcal{MQCL}$.

Le reste de ce chapitre est décomposé comme suit. Nous présentons, dans un premier temps, la problématique principale et l'objectif de notre approche. Dans la section 8.3, nous présentons l'extension en langage du premier ordre du langage de la logique de représentation des préférences que nous utilisons dans notre modèle, et finalement, dans la section 8.3.2, nous décrivons le principe général de notre approche de corrélation d'alertes.

8.2 Notre approche

Tout au long de cette section, nous allons présenter le principe général de notre approche, qui concerne la corrélation d'alertes par l'utilisation d'une logique permettant de modéliser et coder les connaissances et les préférences de l'administrateur réseau.

8.2.1 Problématique

Pour surveiller et protéger un système d'informations de toute vulnérabilité ou activité suspicieuse et anormale, l'administrateur réseau utilise généralement différents systèmes de détection d'intrusions et analyseurs. Le problème majeur est que chaque IDS ou analyseur génère une très large quantité d'alertes. La majorité de ces alertes sont de fausses alertes, des alertes redondantes ou répétitives ou aussi des alertes qui ne représentent pas forcément du risque pour le système, etc.
La tâche de l'administrateur, qui consiste à analyser toutes les alertes générées est réellement très délicate et certainement moins efficace pour différentes raisons :

- Les alertes contiennent des informations moins précises.
- Les alertes sont affichées dans différents formats du fait qu'elles sont générées par différents IDSs.
- L'analyse des alertes est souvent manuelle.
- La gestion des alertes est très longue et coûteuse en terme du temps consommé et de la gravité qui peut se produire en laissant passer des attaques.

8.2.2 Objectifs et nos propositions

L'objectif de notre travail dans le cadre de la corrélation d'alertes, consiste à proposer une approche permettant d'une part, de suivre les objectifs des techniques de corrélation existantes comme par exemple faciliter la tâche de l'administrateur par la réduction des quantités d'alertes générées, et d'autre part, propose l'idée de classifier les alertes en considérant plusieurs éléments comme par exemple les alertes que l'administrateur réseau préférera analyser et voudra ignorer. L'idée principale consiste à prendre en considération non seulement les informations contenues dans les alertes mais également les préférences de l'administrateur réseau ainsi que ses connaissances sur le système d'informations surveillé, sur le fonctionnement de l'IDS ou l'analyseur utilisé, sur les attaques et toute autre information qu'il peut avoir par expérience. Ainsi, nous proposons d'appliquer une des logiques que nous avons développées pour modéliser les préférences et les connaissances de l'administrateur réseau afin de définir uniquement des alertes préférées,

ou plus précisément, des alertes qui satisfont ses connaissances et préférences.

8.3 Choix de la logique à appliquer

Nous avons présenté dans la première partie de cette thèse trois nouvelles logiques : $\mathcal{PQCL}$ qui permet de représenter les préférences des agents ayant différents niveaux d'importance, $\mathcal{QCL}+$ qui permet de représenter les préférences positives et $\mathcal{MQCL}$ qui est une extension simple de la logique $\mathcal{QCL}$. Comme le problème de la corrélation d'alertes consiste en général à réduire le nombre d'alertes générées et comme il n'y a pas besoin de traiter des préférences prioritaires ou positives, la logique $\mathcal{MQCL}$ est largement suffisante pour ce type de problème.

Le langage de ces logiques est une extension du langage propositionnel qui est un langage simple pour la représentation des connaissances. Différents outils d'inférence permettant de traiter de grandes bases de connaissances propositionnelles sont disponibles. Cela est particulièrement vrai pour un certain nombre des solveurs SAT (SAT pour la satisfiabilité des problèmes) qui sont accessibles au public. Toutefois, la logique propositionnelle n'est pas appropriée pour exprimer les connaissances complexes. Elle ne peut traiter que des informations concernant des situations bien particulières, et ne peut pas exprimer les connaissances génériques qui impliquent des variables. Donc, il est important d'avoir un langage riche permettant d'exprimer des informations générales. Ainsi, pour le besoin de notre approche, nous proposons de généraliser et d'étendre le langage de la logique $\mathcal{MQCL}$ en un fragment de la logique du premier ordre afin de pouvoir modéliser les connaissances et les préférences de l'administrateur.

8.3.1 Généralisations et extensions du langage $\mathcal{MQCL}$

Dans cette section, nous allons légèrement étendre le langage $\mathcal{MQCL}$ en un fragment de la logique du premier ordre, c'est-à-dire que toute formule de choix de base ou de choix générale sera étendue et généralisée en une formule du premier ordre universellement quantifiée qui ne nécessite pas de symboles de fonctions. Pour le besoin de notre application, il n'est pas nécessaire de considérer tout le langage de la logique du premier ordre. Nous présentons le nouveau langage, noté $\mathcal{FO}$-$\mathcal{MQCL}$ (pour First Order - Minimal Qualitative Choice Logic) dans ce qui suit :

L'alphabet : le nouveau langage est formé des symboles suivants :
- Un ensemble de variables : $\mathcal{X} = \{x, y, z, ...\}$.
- Un ensemble de constantes : $\mathcal{C} = \{c_1, c_2,...,c_n\}$.
- Un ensemble de symboles de prédicats : $\mathcal{P} = \{p, q, r, ...\}$. Chaque symbole de prédicat p est associé d'un nombre n qui représente l'arité du prédicat, c'est-à-dire le nombre d'arguments associés au prédicat. Si $n = 0$ alors p est tout simplement un symbole propositionnel.

Les termes : un terme est défini comme une constante de l'ensemble $\mathcal{C}$ ou comme une variable de l'ensemble $\mathcal{X}$. Il s'agit d'une définition restreinte de la notion de termes utilisés dans la logique du premier ordre car notre application ne nécessite aucun symbole de fonction.

Les formules : pour le besoin de notre application et pour des raisons de simplicité, nous considérons uniquement des formules du premier ordre universellement quantifiées. Nous donnons d'abord la définition des formules du premier ordre non quantifiées (ou formules atomiques) :

Définition 8.1
L'ensemble de formules du premier ordre non quantifiées est défini à partir des règles suivantes :

(a) Si p est un symbole de prédicat d'arité n et t_1, t_2,...,t_n sont des termes, alors p(t_1, t_2,...,t_n) est une formule du premier ordre non quantifiée.

(b) Si ϕ et ψ sont deux formules du premier ordre non quantifiées, alors $\phi \wedge \psi$, $\phi \vee \psi$, $\neg\phi$, $\phi \vec{\times} \psi$ sont des formules du premier ordre non quantifiées.

(c) Les formules du premier ordre non quantifiées sont obtenues par l'application des règles (a) et (b).

Le langage $\mathcal{FO}$-$\mathcal{MQCL}$ contient des formules propositionnelles universellement quantifiées dans $PROP_{PS}$, utilisées pour exprimer les connaissances d'un administrateur réseau au sujet des alertes qu'il veut analyser ou ignorer, des formules de choix de base universellement quantifiées dans BCF_{PS} pour exprimer les préférences simples et des formules de choix générales universellement quantifiées dans GCF_{PS} pour exprimer les préférences complexes et générales. Ces formules sont définies comme suit :

Définition 8.2
L'ensemble des formules universellement quantifiées (formules $\mathcal{FO}$-$\mathcal{MQCL}$) est obtenu uniquement à partir des règles suivantes :

- *Si ϕ est une formule propositionnelle de $PROP_{PS}$ non quantifiée et {x_1, x_2,..., x_n} est un ensemble de variables utilisées dans ϕ alors $\forall$ x_1,..., $\forall$ x_n ϕ est une formule propositionnelle universellement quantifiée.*
- *Si ψ est une formule de de choix de base non quantifiée et {x_1, x_2,..., x_n} est un ensemble de variables utilisées dans ψ alors $\forall$ x_1,..., $\forall$ x_n ψ est une formule de choix de base universellement quantifiée.*
- *Si ψ' est une formule de de choix générale non quantifiée et {x_1, x_2,..., x_n} est un ensemble de variables utilisées dans ψ' alors $\forall$ x_1,..., $\forall$ x_n ψ' est une formule de choix générale universellement quantifiée.*

Exemple 8.1
Supposons qu'un administrateur réseau préfère analyser toutes les alertes qui sont produites par l'IDS Bro à celles qui sont produites par l'IDS Snort. Nous utilisons le langage $\mathcal{FO}$-$\mathcal{MQCL}$ pour représenter cette préférence qui peut être écrite comme suit :
$\psi = \forall x, \forall y$ *Alert-Bro*(x) $\vec{\times}$*Alert-snort*(y). ψ est une formule de *choix de base universellement quantifiée.*
La formule $\forall x, y, z$ $(Web\text{-}Alerts(x)$ $\vec{\times}$ $Ping\text{-}Alerts(y)) \wedge Scan\text{-}Alerts(z)$ est une formule de choix générale universellement quantifiée.
La formule $\forall x, y$ $\neg Alert\text{-}Bro(x) \vee Alert\text{-}snort(y)$ est une formule propositionnelle universellement quantifiée.

Le langage $\mathcal{FO}$-$\mathcal{MQCL}$ offre une flexibilité pour l'expression des connaissances et des préférences. Cependant, du point de vue du calculatrice, il est préférable de travailler sur un langage propositionnel en vue d'exploiter les outils d'inférence existants. En fait, il est important d'instancier les bases de connaissances du premier ordre en bases de connaissances propositionnelles. Les étapes d'instanciations sont les suivantes :

1. Soient K un ensemble de formules universellement quantifiées, qui ne contiennent pas de symbole de la disjonction ordonnée $\vec{\times}$, et T un ensemble de formules $\mathcal{FO}$-$\mathcal{MQCL}$ universellement quantifiées.
2. Soit $\mathcal{E}(K,T)$ un ensemble de constantes utilisées dans K et T.
3. Soit $\psi = \forall\ x_1,..., \forall\ x_n\ \phi$ une formule $\mathcal{FO}$-$\mathcal{MQCL}$ universellement quantifiée. Définir $\mathcal{D}(\psi)$ comme le domaine associé à ψ. $\mathcal{D}(\psi)$ est le sous ensemble de $\mathcal{E}(K,T)^n$ composé de n-uplets de $\mathcal{E}(K,T)^n$. Il est soit fixé par un expert, soit initialisé par défaut à partir de $\mathcal{E}(K,T)^n$.
4. Définir $\mathcal{I}nstantiation(\psi)$ comme l'ensemble de toutes les formules $\mathcal{FO}$-$\mathcal{MQCL}$ obtenues par le remplacement de (x_1, x_2,..., x_n) dans ψ respectivement par un élément (c_1, c_2,..., c_n) de $\mathcal{D}(\psi)$.
5. Définir $\mathcal{I}nst(K)$ (resp. $\mathcal{I}nst(T)$) comme le résultat d'instanciation de chaque formule de K (resp. de T).

La relation d'inférence des formules universellement quantifiées dans le cadre $\mathcal{FO}$-$\mathcal{MQCL}$ est donnée comme suit :

- Appliquer la définition 3.1 pour transformer toute formule de choix générale universellement quantifiée en une formule de choix de base universellement quantifiée.
- Appliquer les étapes d'instanciation pour chaque formule comme c'est indiqué pour K et T.
- Appliquer la définition 2.3 pour définir le degré de satisfaction de chaque formule obtenue : (item 1) pour toutes les formules BCF ou (item 2) pour les formules propositionnelles.
- Calculer les modèles préférés de la base $K \cup T$ en utilisant la définition 2.8.

Exemple 8.2
Supposons que K contient une connaissance représentée par la formule ϕ_1 telle que :

$$\phi_1 = \forall x\ \neg \mathit{Present\text{-}Ping\text{-}alerts}(x)$$

et T contient une seule préférence représentée par la formule ϕ_2 telle que :

$$\phi_2 = \forall x, \forall y, \forall z\ \mathit{Present\text{-}Web\text{-}alerts}(x) \vee (\mathit{Present\text{-}Scan\text{-}alerts}(y)\ \vec{\times}\ \mathit{Present\text{-}Ping\text{-}alerts}(z)).$$

ϕ_1 est une formule propositionnelle universellement quantifiée permettant de coder la connaissance de l'administrateur réseau qui considère que les alertes $Ping$ ne doivent pas être présentées. ϕ_2 est une formule de choix générale universellement quantifiée indiquant que l'administrateur veut analyser les alertes Web et s'il n'y a pas d'alertes Web, il préfère analyser les alertes $Scan$ à celles de type $Ping$.
Il est clair que l'administrateur est intéressé à analyser uniquement les alertes qui satisfont ses connaissances et ses préférences (c-à-d. les alertes préférées).
Pour savoir quelles sont les alertes préférées, nous suivons les étapes présentées plus haut.

1. *Normalisation de la formule de choix générale universellement quantifiée ϕ_2 :*
   Soit ϕ_2'= $\mathcal{N}_{\mathcal{MQCL}}(\phi_2)$, par l'utilisation de l'item 4-(b) de la définition 3.1, nous avons :
   $\phi_2' = \mathcal{N}_{\mathcal{MQCL}}(\forall x, \forall y, \forall z\ \textit{Present-Web-alerts}(x) \vee (\textit{Present-Scan-alerts}(y) \vec{\times} \textit{Present-Ping-alerts}(z)))$
   $\equiv \forall x, \forall y, \forall z\ (\textit{Present-Web-alerts}(x) \vee\ \textit{Present-Scan-alerts}(y))\ \vec{\times}\ \textit{Present-Ping-alerts}(z)$.
   A ce niveau, $K \cup T$ contient une formule propositionnelle universellement quantifiée ϕ_1 et une formule de choix de base universellement quantifiée ϕ_2'.
2. *Les formules instanciées :* dans cet exemple, nous supposons que nous avons trois alertes différentes : (A_1, A_2 et A_3), donc l'ensemble de constantes est : $\mathcal{E}(K,T) = \{A_1, A_2, A_3\}$. Notons qu'une alerte donnée ne peut pas être au même temps de type Scan, Web et Ping. Les formules instanciées possibles sont :

$$\mathcal{I}nst(K) = \begin{cases} \neg \textit{Present-Ping-alerts}(A_1) & -(1) \\ \neg \textit{Present-Ping-alerts}(A_2) & -(2) \\ \neg \textit{Present-Ping-alerts}(A_3) & -(3) \end{cases}$$

et

$$\mathcal{I}nst(T) = \begin{cases} (\textit{Present-Web-alerts}(A_1) \vee \textit{Present-Scan-alerts}(A_2)) \vec{\times} \textit{Present-Ping-alerts}(A_3) & -(4) \\ (\textit{Present-Web-alerts}(A_1) \vee \textit{Present-Scan-alerts}(A_3)) \vec{\times} \textit{Present-Ping-alerts}(A_2) & -(5) \\ (\textit{Present-Web-alerts}(A_2) \vee \textit{Present-Scan-alerts}(A_1)) \vec{\times} \textit{Present-Ping-alerts}(A_3) & -(6) \\ (\textit{Present-Web-alerts}(A_2) \vee \textit{Present-Scan-alerts}(A_3)) \vec{\times} \textit{Present-Ping-alerts}(A_1) & -(7) \\ (\textit{Present-Web-alerts}(A_3) \vee \textit{Present-Scan-alerts}(A_1)) \vec{\times} \textit{Present-Ping-alerts}(A_2) & -(8) \\ (\textit{Present-Web-alerts}(A_3) \vee \textit{Present-Scan-alerts}(A_2)) \vec{\times} \textit{Present-Ping-alerts}(A_1) & -(9) \end{cases}$$

   Notons aussi qu'avec l'aide des faits concernant les alertes (des détails concernant les faits associés aux alertes sont donnés dans les sections suivantes), certaines formules instanciées sont simplement impossibles. Si par exemple nous avons A_2 est une alerte Web, A_1 est de type Scan et A_3 est un Ping, alors les formules possibles sont (3) et (6).

3. *Les modèles préférés :* dans le tableau 8.1, nous donnons pour chaque formule instanciée (3) dans $\mathcal{I}nst(K)$ et (6) dans $\mathcal{I}nst(T)$ son degré de satisfaction (1, 2) si elle est satisfaite ou non (-) pour toute interprétation.

Present-Web-alerts(A_2)	*Present-Scan-alerts*(A_1)	*Present-Ping-alerts*(A_3)	(3)	(6)
Vrai	Vrai	Vrai	-	1
Vrai	**Vrai**	**Faux**	**1**	**1**
Vrai	Faux	Vrai	-	1
Vrai	**Faux**	**Faux**	**1**	**1**
Faux	Vrai	Vrai	-	1
Faux	**Vrai**	**Faux**	**1**	**1**
Faux	Faux	Vrai	-	2
Faux	Faux	Faux	1	-

TAB. 8.1: Les modèles préférés de $K \cup T$

Les formules (3) et (6) dans $K \cup T$ sont satisfaites par trois interprétations (ligne en gras) et $\textit{Web-alerts}(A_2)$ est vraie dans deux modèles, donc les alertes préférées sont $\textit{Web-alerts}(A_2)$. Cela signifie que l'administrateur va analyser uniquement les alertes Web dans un premier temps.

8.3.2 Application de $\mathcal{FO}$-$\mathcal{MQCL}$ à la corrélation d'alertes

Comme le montre le schéma général de la figure 8.1, notre approche [Benferhat & Sedki 2008a] se décompose en deux parties. La première partie représente les entrées qui contient quatre principaux éléments. La seconde partie représente les sorties de notre modèle.

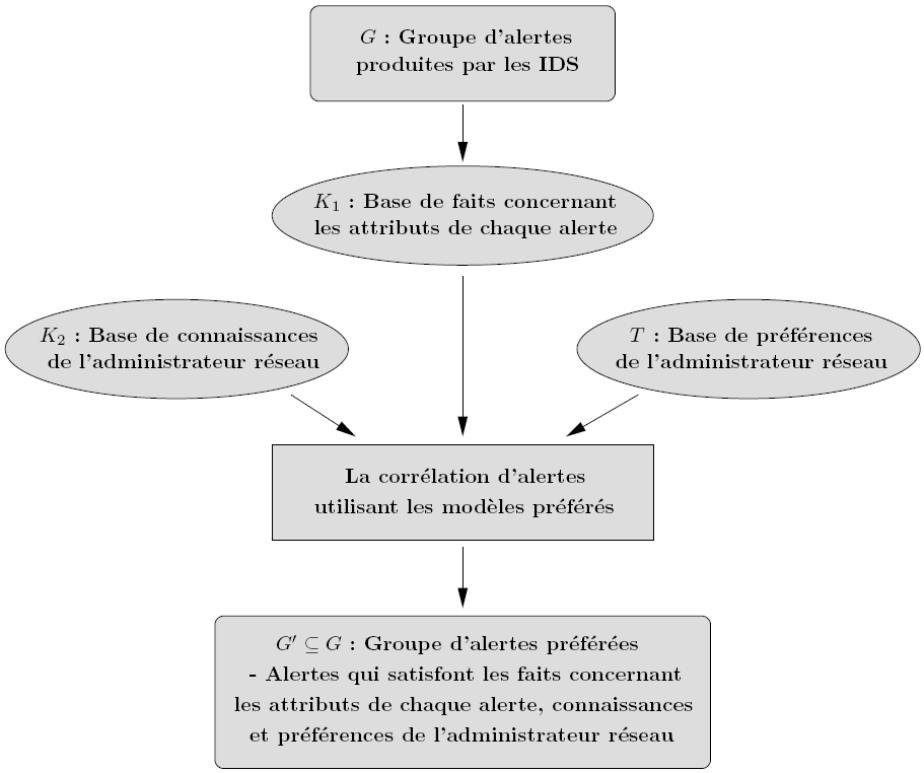

FIG. 8.1: Le schéma général de notre approche

Description des entrées

Les entrées de notre approche concernent quatre principaux éléments : un groupe d'alertes produites par les IDSs, une base de faits représentant différentes informations sur chaque alerte, une base de connaissances de l'administrateur réseau ainsi qu'une base de préférences de l'administrateur. Ces éléments sont décrits dans ce qui suit :

1. **Groupe d'alertes G produites par les IDSs :** chaque alerte est caractérisée par un ensemble d'attributs nommés "*attributs de base*". Exemples d'attributs de base sont : Timestamp, Signature d'alerte (Sid), message associé à l'alerte, Protocole, les adresses IP source et destination, les ports source et destination, TTL (Time To Live), le champ identification (ID), etc. L'exemple de la figure 8.2 présente quelques informations qu'une alerte peut contenir.

   Exemples d'attributs associés à l'alerte de la figure 8.2 sont :
   - *Timestamp* : 03/08-16 :40 :14.086388,
   - *Signature* : 1156,
   - *Message associé à l'alerte* : WEB-MISC apache directory disclosure attempt,
   - *Protocole* : TCP,

03/08-16 :40 :14.086388,1,1156,10,WEB-MISC apache directory disclosure attempt, TCP,199.174.194.16,1366,172.16.114.50,80,0 :C0 :4F :A3 :58 :23,0 :10 :7B :38 :46 :32, 0x5EA,***A****,0xF353B4E4,0xA4BEF48D,,0x7D78,63,0,986,1500,225300,,,,

FIG. 8.2: Exemple d'une alerte Snort

- *Les adresses IP* : 199.174.194.16 et 172.16.114.50,
- *Les ports* : 1366 et 80 (http),
- *TTL* : 63.

2. **Base de faits :** chaque attribut qui caractérise une alerte donnée est représenté par un symbole de prédicat que l'on appelle aussi "*fait*". L'ensemble des faits est représenté dans une base, notée par K_1.

   Exemple 8.3
   Supposons que le groupe d'alertes G contient une seule alerte identifiée par id_1. Supposons que les attributs concernant cette alerte sont : l'IDS qui a produit l'alerte est Snort, le protocole utilisé par l'attaquant est TCP et la classe de l'attaque est DoS. Ces faits seront représentés par $K_1 = \{IDS(id_1, Snort), Protocol(id_1, TCP), Class(id_1, DoS)\}$ qui indique que l'alerte id_1 est caractérisée par les attributs IDS $Snort$, le protocole TCP et la classe d'attaque DoS. Notons en général que certains attributs ne sont pas informés (connus) par l'IDS qui génère les alertes.

   Types de faits : nous distinguons deux types de faits :

   (a) **Faits informés par l'IDS :** ces faits sont directement définis à partir des attributs de base des alertes générées.

   Exemple 8.4
   $Protocol(A_1, TCP)$ est un exemple d'un fait direct qui indique que l'attribut protocole de l'alerte A_1 prends la valeur TCP.
   $Sid(A_2, 1156)$ est un exemple d'un autre fait qui renseigne sur la signature de l'alerte A_2.

   (b) **Faits non informés par l'IDS :** ce type de faits concerne les attributs non connus ou non informés par l'IDS qui a généré les alertes. La direction de l'alerte est un exemple de fait non informé directement par l'IDS. Il se base sur les adresses IP source et destination. L'information que représente ce fait, permet de connaître la direction des alertes produites (inbound, outbound, inside).

   Un autre exemple de ce type de faits est la classe de l'attaque. Selon les signatures des alertes connues, nous pouvons déterminer le fait "*Classe de l'alerte*". Le tableau 8.2 décrit quelques signatures d'alertes et la classe d'attaque correspondante.
   Dans l'exemple de la figure 8.2, le fait Sid (1156) de l'alerte ou $Message$(*WEB-MISC apache directory disclosure attempt*) permettent de déterminer la classe de l'activité anormale que l'alerte décrive qui est de type DoS. Nous pouvons également vérifier cela dans la règle snort qui a permet de générer cette alerte (voir figure 5.3 du chapitre 5).

Classe de l'alerte	Sid (Signature de l'alerte)
DoS	527, 151, 1257, 368, 1156, 1418, 1420, 1421, 450, 2003
Probe	365, 407, 409, 503, 973, 1668, 1648, 2473, 2470, 2467, 832,
R2L	553, 335, 491, 718, 652, 1104, 648, 1417, 498, 1748, 1147,
U2R	716, 718, 1292, 1842,

TAB. 8.2: Exemples de classes d'attaques associées aux signatures des alertes

3. **Connaissances de l'administrateur réseau :** l'administrateur réseau peut fournir des connaissances concernant le fonctionnement de son système, les IDSs ou les analyseurs qu'il utilise ou toute autre information qui lui permettra de faciliter l'analyse des alertes, etc. Ces connaissances seront représentées dans une base de connaissances, notée par K_2. La base K_2 contient un ensemble de formules propositionnelles universellement quantifiées (c-à-d. formules qui ne contiennent pas le symbole de préférences $\vec{\times}$).
   Ce type d'informations est important car par expérience, l'administrateur réseau peut apporter beaucoup de connaissances sur différentes catégories d'attaques. Des attaques connues, telles que, ftpwrite (accès au fichier .rhost), geuss (tentative de deviner un login et un mot de pass) peuvent être facilement détectées, si par exemple, l'administrateur tient compte des signatures (Sid) des alertes affichées lors de la présence de ce type d'attaques. Un exemple d'une connaissance simple pour présenter à l'administrateur des alertes qui décrivent les attaques ftpwrite et guess est : $\forall x, y\ Sid(x, 553) \wedge Sid(y, 718) \wedge \rightarrow Present\text{-}alert(x) \wedge Present\text{-}alert(y)$.
4. **Préférences de l'administrateur réseau :** l'administrateur réseau peut exprimer ses préférences sur ce qu'il veut analyser en premier et ce qu'il souhaite ignorer. Si les préférences sont d'une forme simple, alors elles seront représentées par des formules de choix de base universellement quantifiées, si elles sont d'une forme complexe, alors elles seront représentées par des formules de choix générales universellement quantifiées. L'ensemble des préférences constitue une base de préférences, notée par T.

   Exemple 8.5
   Soit ϕ une formule de choix de base universellement quantifiée :
   $\phi = \forall x, \forall y\ IDS(x, Snort) \wedge Class(x, R2L) \wedge IDS(y, Bro) \wedge Class(y, DoS) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$.
   Intuitivement, cette formule signifie que, si une alerte x issue de l'IDS Snort concerne une attaque R2L, et si une alerte y issue de l'IDS Bro concerne l'attaque DoS, alors l'administrateur réseau préfère analyser en premier l'alerte x, et par la suite l'alerte y.

Sorties de notre modèle

Les sorties de notre approche, concernent un sous ensemble $G' \subset G$. Plus précisément, un sous ensemble d'alertes dans G qui seront en premier présentées à l'administrateur.
L'objectif de notre technique de corrélation d'alertes, consiste à présenter en premier uniquement des alertes qui satisfont les connaissances et les préférences de l'administrateur réseau, c'est-à-dire les alertes préférées. Si besoin, des alertes préférées en second degré seront également présentées, et ainsi de suite.
Pour sélectionner des alertes préférées, nous avons besoin de traiter les alertes produites, les

connaissances et les préférences de l'administrateur réseau, et les coder dans le cadre de notre logique. En particulier, nous avons besoin de :

- Extraire un ensemble de faits K_1 de l'ensemble des alertes produites. Il est à préciser, qu'il est suffisant d'extraire uniquement des faits qui permettent de représenter les caractéristiques des alertes spécifiées dans les connaissances et les préférences de l'administrateur.
- Représenter chaque connaissance de l'administrateur réseau par une formule propositionnelle universellement quantifiée. L'ensemble de connaissances seront représentées dans la base K_2.
- Représenter chaque préférence simple, par une formule de choix de base universellement quantifiée, et les préférences complexes par des formules de choix générales universellement quantifiées. L'ensemble de préférences seront représentées dans la base T.
- Utiliser $\mathcal{N}_{\mathcal{MQCL}}$ (Définition 3.1) pour normaliser les formules de choix générales si elles existent.
- Appliquer les étapes d'instanciation selon l'ensemble des constantes de $\mathcal{E}(K_1 \cup K_2, T)$.
- Sélectionner uniquement des formules instanciées qui concernent les faits considérés.
- Calculer le degré de satisfaction de chaque formule instanciée (utilisation de la relation d'inférence $\mathcal{MQCL}$).
- Déterminer les modèles préférés.
- Finalement, définir G' comme l'ensemble d'alertes qui sont vraies dans tous les modèles préférés.

Un exemple détaillé

Supposons que l'administrateur réseau a, à analyser l'ensemble des alertes données dans le tableau 8.3. Ce tableau contient 8 alertes générées par l'IDS Snort sur un trafic réseau.

Alertes	Timestamp	Sid	Message	Protocole	IPsrc	Portsrc	IPdst	Portdst	TTL
A_1	3/8/16/39/12.024	1156	WEB-MISC apache-directory	TCP	199.174.194.16	1028	172.16.114.50	80	62
A_2	3/8/16/39/12.0267	1156	WEB-MISC apache-directory	TCP	199.174.194.16	1028	172.16.114.50	80	62
A_3	3/8/16/39/12.030	1156	WEB-MISC apache-directory	TCP	199.174.194.16	1028	172.16.114.50	80	62
A_4	3/11/17/50/16.384	469	ICMP PING	ICMP	207.103.80.104	8 (Type)	172.16.114.50	0 (Code)	63
A_5	4/1/02/19/43.008	1893	SNMP missing	UDP	172.16.0.1	1336	207.181.92.211	161	63
A_6	4/1/02/28/22.802	895	WEB-CGI	TCP	172.16.112.207	6677	143.166.224.44	0	63
A_7	3/8/15/01/13.344	1648	*WEB-CGI* perl.exe	TCP	206.48.44.18	1061	172.16.112.100	80	127
A_8	3/9/15/43/31.968	2925	INFO web bug 1x1 gif attempt	TCP	199.95.74.90	80	172.16.113.105	4671	112

Tab. 8.3: Groupe d'alertes levées par l'IDS Snort

Nous supposons que l'administrateur réseau fournit l'ensemble des connaissances et des préférences suivant pour sélectionner uniquement des alertes préférées :

Préférences de l'administrateur :

- L'administrateur réseau préfère des alertes identifiées par le protocole TCP aux alertes identifiées par le protocole UDP à celles qui sont identifiées par le protocole $ICMP$. Plus précisément, si x, y, z sont trois alertes et si le protocole de x (resp. y, z) est TCP (resp. UDP, ICMP) alors il est préférable de présenter en premier x, y en second et z par la suite.
- Il préfère les alertes entrantes à celles qui sont sortantes. Les alertes sont considérées "entrantes" si les adresses IP source des événements qu'elles décrivent ne font pas partie du réseau. Elles sont "sortantes" si leurs adresses IP source font partie du réseau, mais pas les adresses IP destination.

Connaissances de l'administrateur :

- L'administrateur réseau souhaite ignorer toutes alertes "INFO web bug 1x1 gif attempt". Par cette connaissance, l'administrateur réseau ne s'intéresse pas à analyser toutes les alertes dont le message qui leur sont associé est "INFO web bug 1x1 gif attempt".
- Il veut également ignorer toutes les alertes redondantes. Nous considérons des alertes redondantes, celles qui ont les mêmes attributs (Sid, protocole, adresses IP source et destination, ports source et destination) et différents timestamp. Donc, il s'agit de supprimer toutes les alertes ayant les mêmes attributs cités et garder que l'alerte ayant le plus petit timestamp. Dans le tableau 8.3, les alertes A_1, A_2 et A_3 ont les mêmes caractéristiques, mais leur timestamp sont différents. Le résultat attendu de notre modèle est qu'il considère A_2 et A_3 des alertes redondantes car A_1 a le plus petit timestamp parmi ces trois alertes.

Les symboles de prédicats de notre langage, sont des prédicats associés aux attributs des alertes : Timestamp, Sid, Portdst,..., et :

- $Differ(A_i, A_j)$ signifie que les alertes A_i et A_j ont différentes identités.
- $Present\text{-}alert(id)$ lorsqu'il prend la valeur vrai, signifie que l'alerte doit être présentée à l'administrateur réseau.

Nous avons l'ensemble des alertes produites par l'IDS Snort, les connaissances et les préférences de l'administrateur réseau. Les étapes nécessaires pour sélectionner les alertes préférées sont présentées ci-dessous :

1) Extraire la base de faits K_1

Pour extraire les faits associés aux alertes, nous sélectionnons uniquement, ceux qui sont spécifiés dans les connaissances et les préférences de l'administrateur réseau. Les faits nécessaires sont :

- Les faits informés par l'IDS, sont directement extraits à partir des alertes générées (voir le tableau 8.3). $Protocol(A_1, TCP)$ est un exemple de ce type de faits, qui concerne le protocole TCP de l'alerte A_1.

- Les faits non informés par l'IDS qui doivent être définis sont :
  A partir la deuxième préférence, indiquant que l'administrateur préfère des alertes entrantes aux alertes sortantes, nous déduisons qu'un fait qui renseigne sur la direction des alertes et non informé directement par l'IDS doit être défini. Nous appelons ce fait

"*Direction*", qui prend soit "inbound" soit "outbound".
Dans cet exemple, les alertes entrantes sont des alertes dont l'adresse IP source est différente de "172.16.x.y" et l'adresse IP destination est égale à "172.16.x.y". Le fait $Direction(A_1, inbound)$ renseigne sur la direction de l'alerte qui est de type "inbound" (alerte entrante). Les alertes sortantes sont des alertes dont l'adresse IP source est égale à "172.16.x.y" et l'adresse IP destination est différente de "172.16.x.y". Le fait $Direction(A_6, outbound)$ indique que l'alerte A_6 est de type outbound (alerte sortante).

La deuxième connaissance de l'administrateur indique qu'il veut ignorer toutes les alertes redondantes. Pour pouvoir identifier les alertes redondantes, des faits qui renseignent sur la similarité des attributs (Sid, protocole, adresses IP source et destination, ports source et destination) entre les alertes doivent être définis. Un autre fait qui indique l'alerte ayant le plus petit timestamp parmi les alertes similaires doit être également défini. Ces faits sont les suivants : *SameIPsrc*, *SameIPdst*, *SamePortsrc*, *SamePortdst*, *SameSid* qui correspondent respectivement aux alertes ayant les mêmes attributs (Sid, protocole, adresses IP source et destination, ports source et destination). Le fait *Timestamp-samaller-than* correspond au plus petit timestamp. Dans notre exemple, ces faits sont appliqués aux alertes A_1, A_2, A_3, qui ont les mêmes attributs notés et l'alerte A_1 a le plus petit timestamp que les alertes A_2 et A_3. A_2 et A_3 sont considérées des alertes répétitives ou redondantes. Le fait "$Timestamp\text{-}Smaller\text{-}than(A_1, A_2)$" signifie que le timestamp de l'alerte A_1 est inférieur au timestamp de l'alerte A_2.

Notons que nous considérons uniquement les faits qui apparaissent dans les connaissances et les préférences de l'administrateur. L'ensemble des faits est donné comme suit :

$$K_1 = \begin{cases} Protocol(A_1, TCP), Protocol(A_2, TCP), Protocol(A_3, TCP), \\ Protocol(A_4, ICMP), Protocol(A_5, UDP), Protocol(A_6, TCP), \\ Protocol(A_7, TCP), Protocol(A_8, TCP), Direction(A_1, inbound), \\ Direction(A_2, inbound), Direction(A_3, inbound), Direction(A_4, inbound), \\ Direction(A_5, outbound), Direction(A_6, outbound), Direction(A_7, inbound), \\ Direction(A_8, inbound), Message(A_8, INFO\ web\ bug\ 1x1\ gif\ attempt), \\ SameIPsrc(A_1, A_2), SameIPsrc(A_1, A_3), SameIPsrc(A_2, A_3), \\ SameIPdst(A_1, A_2), SameIPdst(A_1, A_3), SameIPdst(A_2, A_3), \\ SamePortsrc(A_1, A_2), SamePortsrc(A_1, A_3), SamePortsrc(A_2, A_3), \\ SamePortdst(A_1, A_2), SamePortdst(A_1, A_3), SamePortdst(A_2, A_3), \\ SameSid(A_1, A_2), SameSid(A_1, A_3), SameSid(A_2, A_3), Timestamp\text{-}Smaller\text{-}than(A_1, A_3), \\ Timestamp\text{-}Smaller\text{-}than(A_2, A_3), Timestamp\text{-}Smaller\text{-}than(A_1, A_2) \end{cases}$$

2) Modéliser les connaissances de l'administrateur

Dans ce qui suit, nous modélisons les connaissances de l'administrateur réseau données dans l'exemple. Nous rappelons d'abord chaque connaissance et nous donnons par la suite, la formule logique correspondante.

L'administrateur réseau, ne souhaite pas analyser toutes les alertes "$INFO\ web\ bug\ 1x1\ gif\ attempt$", cela signifie que notre modèle ne doit pas présenter ce type d'alertes à l'administrateur. Cette connaissance est modélisée par la formule ϕ_1 comme suit :

- $\phi_1 = \forall x\ Message(x, INFO\ web\ bug\ 1x1\ gif\ attempt) \rightarrow \neg\ Present\text{-}alert(x)$.

La deuxième connaissance qui indique qu'il veut également ignorer toutes les alertes redondantes, est codée par la formule ϕ_2 comme suit :

- $\phi_2 = \forall x, \forall y\ SameIPsrc(x,y) \wedge SameIPdst(x,y) \wedge SamePortsrc(x,y) \wedge SamePortdst(x,y) \wedge SameSid(x,y) \wedge Timestamp\text{-}Smaller\text{-}than(x,y) \wedge Differ(x,y) \rightarrow Present\text{-}Alert(x) \wedge \neg Present\text{-}Alert(y)$.

Il est à préciser que nous supposons que si deux alertes sont différentes, alors leurs timestamp sont différents.

3) Modéliser les préférences de l'administrateur

La préférence "L'administrateur réseau préfère des alertes identifiées par le protocole TCP aux alertes identifiées par le protocole UDP à celles qui sont identifiées par le protocole $ICMP$" est modélisée par la formule ϕ_3 suivante :

- $\phi_3 = \forall x, \forall y, \forall z\ Protocol(x, TCP) \wedge Protocol(y, UDP) \wedge Protocol(z, ICMP) \wedge Differ(x,y,z) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y) \vec{\times} Present\text{-}alert(z)$.

La préférence "Il préfère les alertes entrantes à celles qui sont sortantes" est codée par la formule ϕ_4 comme suit :

- $\phi_4 = \forall x, \forall y\ Direction(x, inbound) \wedge Direction(y, outbound) \wedge Differ(x,y) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$.
  - $\forall x,\ \forall y\ Direction(x, inbound) \wedge Direction(y, outbound) \wedge Differ(x,y) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$.

4) Définir $\mathcal{I}nst(T)$ et $\mathcal{I}nst(K_2)$ (instanciation des formules de T et de K_2)

Pour définir $\mathcal{I}nst(T)$ et $\mathcal{I}nst(K_2)$, une méthode directe, consiste à considérer toutes les différentes possibilités selon les différentes valeurs des alertes. Par exemple, pour ϕ_3, nous avons besoin de considérer toutes les paires (x, y, z) de $\{A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8\}^3$. Cependant, avec l'aide des faits de la base K_1, certaines instanciations sont impossibles. Par exemple, dans ϕ_3, il n'est pas nécessaire de considérer (A_4, A_5, A_1) car cela induit à : $Protocol(A_4, TCP) \wedge Protocol(A_5, UDP) \wedge Protocol(A_1, ICMP) \rightarrow Present\text{-}alert(A_4) \vec{\times} Present\text{-}alert(A_5) \vec{\times} Present\text{-}alert(A_1)$. Alors que le fait $Protocol(A_1, ICMP)$ ne sera jamais appliqué et donc il peut être suprimé de la base de faits. Étant données, ces observations, $\mathcal{I}nst(K_2)$ et $\mathcal{I}nst(T)$ sont données dans ce qui suit :

$$\mathcal{I}nst(K_2) = \begin{cases} \psi_1 = Message(A_8, INFO\ web\ bug\ 1x1\ gif\ attempt) \rightarrow \neg Present\text{-}alert(A_8) \\ \psi_2 = SameIPsrc(A_1, A_2) \wedge SameIPdst(A_1, A_2) \\ \wedge SamePortsrc(A_1, A_2) \wedge SamePortdst(A_1, A_2) \\ \wedge SameSid(A_1, A_2) \wedge Timestamp\text{-}Smaller\text{-}than(A_1, A_2) \\ \rightarrow Present\text{-}alert(A_1) \wedge \neg Present\text{-}alert(A_2) \\ \psi_3 = SameIPsrc(A_1, A_3) \wedge SameIPdst(A_1, A_3) \\ \wedge SamePortsrc(A_1, A_3) \wedge SamePortdst(A_1, A_3) \\ \wedge SameSid(A_1, A_3) \wedge Timestamp\text{-}Smaller\text{-}than(A_1, A_3) \\ \rightarrow Present\text{-}alert(A_1) \wedge \neg Present\text{-}alert(A_3) \\ \psi_4 = SameIPsrc(A_2, A_3) \wedge SameIPdst(A_2, A_3) \\ \wedge SamePortsrc(A_2, A_3) \wedge SamePortdst(A_2, A_3) \\ \wedge SameSid(A_2, A_3) \wedge Timestamp\text{-}Smaller\text{-}than(A_2, A_3) \\ \rightarrow Present\text{-}alert(A_2) \wedge \neg Present\text{-}alert(A_3). \end{cases}$$

et

$$\mathcal{I}nst(T) = \begin{cases} \psi_5 = Protocol(A_1, TCP) \wedge Protocol(A_5, UDP) \wedge Protocol(A_4, ICMP) \\ \rightarrow Present\text{-}alert(A_1) \vec{\times} Present\text{-}alert(A_5) \vec{\times} Present\text{-}alert(A_4) \\ \psi_6 = Protocol(A_2, TCP) \wedge Protocol(A_5, UDP) \wedge Protocol(A_4, ICMP) \\ \rightarrow Presente\text{-}alert(A_2) \vec{\times} Present\text{-}alert(A_5) \vec{\times} Present\text{-}alert(A_4) \\ \psi_7 = Protocol(A_3, TCP) \wedge Protocol(A_5, UDP) \wedge Protocol(A_4, ICMP) \\ \rightarrow Present\text{-}alert(A_3) \vec{\times} Present\text{-}alert(A_5) \vec{\times} Present\text{-}alert(A_4) \\ \psi_8 = Protocol(A_6, TCP) \wedge Protocol(A_5, UDP) \wedge Protocol(A_4, ICMP) \\ \rightarrow Present\text{-}alert(A_6) \vec{\times} Present\text{-}alert(A_5) \vec{\times} Present\text{-}alert(A_4) \\ \psi_9 = Protocol(A_7, TCP) \wedge Protocol(A_5, UDP) \wedge Protocol(A_4, ICMP) \\ \rightarrow Present\text{-}alert(A_7) \vec{\times} Present\text{-}alert(A_5) \vec{\times} Present\text{-}alert(A_4) \\ \psi_{10} = Protocol(A_8, TCP) \wedge Protocol(A_5, UDP) \wedge Protocol(A_4, ICMP) \\ \rightarrow Present\text{-}alert(A_8) \vec{\times} Present\text{-}alert(A_5) \vec{\times} Present\text{-}alert(A_4) \\ \psi_{11} = Direction(A_1, inbound) \wedge Direction(A_5, outbound) \\ \rightarrow Present\text{-}alert(A_1) \vec{\times} Present\text{-}alert(A_5) \\ \psi_{12} = Direction(A_1, inbound) \wedge Direction(A_6, outbound) \\ \rightarrow Present\text{-}alert(A_1) \vec{\times} Present\text{-}alert(A_6) \\ \psi_{13} = Direction(A_2, inbound) \wedge Direction(A_5, outbound) \\ \rightarrow Present\text{-}alert(A_2) \vec{\times} Present\text{-}alert(A_5) \\ \psi_{14} = Direction(A_2, inbound) \wedge Direction(A_6, outbound) \\ \rightarrow Present\text{-}alert(A_2) \vec{\times} Present\text{-}alert(A_6) \\ \psi_{15} = Direction(A_3, inbound) \wedge Direction(A_5, outbound) \\ \rightarrow Present\text{-}alert(A_3) \vec{\times} Present\text{-}alert(A_5) \\ \psi_{16} = Direction(A_3, inbound) \wedge Direction(A_6, outbound) \\ \rightarrow Present\text{-}alert(A_3) \vec{\times} Present\text{-}alert(A_6) \\ \psi_{17} = Direction(A_4, inbound) \wedge Direction(A_5, outbound) \\ \rightarrow Present\text{-}alert(A_4) \vec{\times} Present\text{-}alert(A_5) \\ \psi_{18} = Direction(A_4, inbound) \wedge Direction(A_6, outbound) \\ \rightarrow Present\text{-}alert(A_4) \vec{\times} Present\text{-}alert(A_6) \\ \psi_{19} = Direction(A_7, inbound) \wedge Direction(A_5, outbound) \\ \rightarrow Present\text{-}alert(A_7) \vec{\times} Present\text{-}alert(A_5) \\ \psi_{20} = Direction(A_7, inbound) \wedge Direction(A_6, outbound) \\ \rightarrow Present\text{-}alert(A_7) \vec{\times} Present\text{-}alert(A_6) \\ \psi_{21} = Direction(A_8, inbound) \wedge Direction(A_5, outbound) \\ \rightarrow Present\text{-}alert(A_8) \vec{\times} Present\text{-}alert(A_5) \\ \psi_{22} = Direction(A_8, inbound) \wedge Direction(A_6, outbound) \\ \rightarrow Present\text{-}alert(A_8) \vec{\times} Present\text{-}alert(A_6) \end{cases}$$

5) Calculer les modèles préférés

Le résultat attendu de notre modèle, doit répondre aux objectifs et souhaits de l'administrateur (ses connaissances et ses préférences). C'est-à-dire que dans le groupe G', que notre modèle est censé de donner, les alertes préférées (avec le degré 1) doivent satisfaire les critères suivants :

- aucune alerte de type ICMP ou UDP,
- toutes les alertes préférées sont de type TCP

– parmi les alertes TCP préférées, aucune, ne doit avoir "$INFO\ web\ bug\ 1x1\ gif\ attempt$" comme message d'alerte,
– aucune alerte outbound (sortante),
– aucune alerte redondante, c'est-à-dire que G' ne doit pas contenir deux alertes ayant les mêmes attributs (Sid, IP source et destination, Ports source et destination) et différents timstamp.

Pour calculer l'ensemble des modèles préférés de $K_2 \cup T$, c'est-à-dire pour présenter les alertes préférées, nous donnons le degré de satisfaction de toutes les formules instanciées dans $\mathcal{I}nst(K_2)$ et $\mathcal{I}nst(T)$, c'est-à-dire les formules ψ_1, ψ_2, ψ_3, ψ_4, ψ_5, ψ_6, ψ_7, ψ_8, ψ_9, ψ_{10}, ψ_{11}, ψ_{12}, ψ_{13}, ψ_{14}, ψ_{15}, ψ_{16}, ψ_{17}, ψ_{18}, ψ_{19}, ψ_{20}, ψ_{21}, ψ_{22} pour toute interprétation.
Nous concluons, que les modèles préférés contiennent $\{Present\text{-}alert(A_1), Present\text{-}alert(A_7)\}$. Donc, $G' = \{A_1, A_7\}$ est le groupe d'alertes qui sera présenté en premier à l'administrateur.

8.4 Application sur les données DADDi et les données DARPA'99

Nous présentons dans cette section quelques résultats d'application sur deux types d'alertes générées par l'IDS Snort : des alertes concernant les données DADDi (Dependable Anomaly Detection with Diagnosis) et des alertes concernant les données DARPA'99 [Darpa 1999]. Nous n'avons pas la chance d'avoir de vraies règles expertes. Concernant les alertes DADDi par exemple, les membres du projet ont exprimés leur préférences pour l'analyse des attaques HTTP, mais sans ignorer d'autres comme les SQL Slamer ou les Scan par exemple. Ainsi, nous avons choisi quelques règles tout en essayant de satisfaire leurs besoins. Les données DADDi sont décrites ci-après.

8.4.1 Description des données DADDi

Le traffic brut concernant ces données, a été collecté pendant plusieurs jours dans le cadre du projet DADDi (Dependable Anomaly Detection with Diagnosis), au sein du réseau d'un campus universitaire avec le sniffer TCPDump. Le trafic mis a notre disposition, est celui capturé entre 24 mai 2007 et le 11 juin 2007 d'une capacité d'environs 100 Go (sauvegardé dans 100 fichiers). Il contient du trafic a priori normal plus un petit nombre d'attaques, tels que, les ping ICMP, Portscan, slammer (worm propagation). La quantité des données que contient ce trafic est très grande, et leur analyse avec Snort a généré énormément d'alertes. Pour l'application de ces données, nous n'utilisons pas tout le trafic, nous considérons uniquement le trafic des 15 premiers fichiers (fichiers [1-5], [6-10] et [11-15]). Les statistiques concernant les alertes générées sont présentées dans la figure 8.3. Le tableau 8.4 présente quelques exemples d'alertes générées dans les données DADDi.

8.4.2 Résultats de la corrélation d'alertes sur les données DADDi

Les éléments importants pour notre approche sont les connaissances et les préférences de l'administrateur réseau. Nous considérons les connaissances et les préférences suivantes :

1) Connaissances de l'administrateur réseau

- $\forall x, \forall y\ SameIPsrc(x,y) \wedge SameIPdst(x,y) \wedge SamePortsrc(x,y) \wedge SamePortdst(x,y) \wedge SameSid(x,y) \wedge Timestamp\text{-}Smaller\text{-}than(x,y) \wedge Differ(x,y) \rightarrow Present\text{-}Alert(x)$

FIG. 8.3: Statistiques des alertes DADDi

Type d'alertes	Message associé à l'alerte
TCP	- WEBMISC robots.txt access
	- (portscan) TCP Portsweep
	- (http_inspect) BARE BYTE UNICODE ENCODING
	- POLICY Google Desktop activity
UDP	- MYSQL Worm propagation attempt
	- RPC portmap NFS request UDP
	- SHELLCODE x86 NOOP
ICMP	- ICMP PING NMAP
	- ICMP TimeToLive Exceeded in Transit
	- ICMP Destination Unreachable Port Unreachable

TAB. 8.4: Exemples d'alertes dans les données DADDi

$\wedge\neg Present\text{-}Alert(y)$.
Par cette connaissance, l'administrateur réseau, veut supprimer toutes les alertes redondantes (voir des explications sur ce point dans l'exemple détaillé).

- $\forall x\ Protocol(x, ICMP) \wedge Type(x, 8) \wedge Code(x, 0) \rightarrow Present\text{-}alert(x)$.
  Il veut représenter les alertes dont le type est "8" et le code est "0" (alertes Ping).

- $\forall x\ Protocol(x, UDP) \wedge Port(x, 1434) \rightarrow Present\text{-}alert(x)$.

  L'administrateur réseau veut analyser toutes les alertes "UDP" qui décrivent les paquets envoyés vers le port (1334). Ce type d'alertes représentent la vulnérabilité "SQL Slammer" qui est basée sur un débordement de buffer dans le service de résolution de SQL serveur.

2) Préférences de l'administrateur réseau

- $\forall x,\ \forall y,\ Service(x, http) \wedge Service(y, other) \wedge Differ(x, y) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$.
  Cette préférence signifie que l'administrateur préfère analyser les alertes "http" à toutes les autres.

- $\forall x, \forall y, \forall z\ Direction(x, inbound) \wedge Direction(y, outbound) \wedge Direction(z, inside)\ Differ(x, y, z) \rightarrow (Present\text{-}alert(x) \vee Present\text{-}alert(y)) \vec{\times} Present\text{-}alert(z)$.
  La préférence modélisée par la formule ϕ_2 indique que l'administrateur préfère analyser les alertes "inbound" et "outbound" en premier, les alertes "inside" en second.

- $\forall x, \forall y\ Hote(x, Spider) \wedge Hote(y, Other) \wedge Message(x,$ "WEBMISC robots.txt access") $\wedge\ Message(y,$ "WEBMISC robots.txt access") $\wedge\ Differ(x, y) \rightarrow Present\text{-}alert(y) \vec{\times} \neg\ Present\text{-}alert(x)$.

  Pour spécifier aux programmes robots[13] utilisés par les moteurs de recherche quelles sont les pages à ne pas indexer, les administrateurs de sites Web indiquent ces pages dans un fichier spécial "Robot.txt" que les robots accèdent, pour éviter de visiter et indexer les pages en question. Ce fichier est placé à la racine du site Web. Des utilisateurs mal intentionnés accèdent souvent aux fichiers "Robot.txt" car ils pourraient y trouver des informations importantes concernant des fichiers cachés, des arborescences de répertoires secrètes ou toute autre information que les administrateurs auraient laissée par mégarde ou imprudence. Ainsi, Snort lève une alerte à chaque fois que le fichier "Robot.txt" est accédé. Etant donné le nombre de robots et la nature répétitive de l'indexation des contenus des sites Web, le nombre d'alertes générées quotidiennement concernant l'accès au fichier "Robot.txt" est très important (dans les fichiers [1-5] par exemple, nous trouvons 295 alertes). Pour réduire le nombre de ces alertes, ces dernières peuvent regroupées selon les hôtes sources ayant accédé au fichier "Robot.txt". De plus, l'administrateur peut ignorer les alertes où les hôtes sources sont connus et de confiance (comme les moteurs de recherches connus). Cette préférence, indique que notre modèle ne doit pas présenter les alertes qui représentent des accès au fichier "Robot.txt" dont les hôtes sources appartiennent au groupe *Spider* (c-à-d. moteurs de recherches connus tels que google et yahoo). Par contre, les alertes qui représentent des accès au fichier "Robot.txt" dont les hôtes sources appartiennent à un groupe autre que *Spider* doivent être représentées. Ainsi, nous utilisons le fait *Spider* qui regroupe des hôtes sources des moteurs de recherche connus, et *Other* les autres hôtes sources.

Les résultats d'application sont affichés dans le tableau 8.5.

Nous remarquons que le taux de réduction des alertes est important, même si nous avons utilisé plusieurs règles (connaissances et préférences). Le groupe G' contient plusieurs alertes suspectes qui nécessitent l'analyse telles que les "SQL Slammer", "Ping", "accès au fichier Robot.txt". Notons qu'il existe une grande quantité d'alertes "Ping", notamment des alertes redondantes, environs (48592 alertes) dans les fichiers [1-5] par exemple. Cela signifie que si l'administrateur réseau veut encore réduire le nombre d'alertes, il peut changer ses préférences en considérant l'analyse des alertes "Ping" moins importante que les autres. D'ailleurs, lorsque nous

[13] Programmes d'indexation de pages Web

	# d'alertes	# d'alertes préférées	Taux de réduction
Fichiers [1-5]	76380	24953	67.33%
Fichiers [6-10]	77825	31090	60.05%
Fichiers [11-15]	97788	36615	62.55%
Total	**251993**	**92658**	**63.31%**

TAB. 8.5: Les alertes préférées dans les données DADDi

avons retiré la connaissance concernant la présentation des alertes Ping, le nombre d'alertes préférées est égal à 9671 dans les alertes des fichiers [1-5]. Ce qui est important aussi, est qu'il n'y a pas d'alertes répétitives dans le groupe des alertes préférés, alors qu'elles existent en quantités très importantes. Dans les fichiers [1-5] par exemple, nous trouvons environs 45571 alertes redondantes, dont la majorité sont des ping.

8.4.3 Résultats de la corrélation d'alertes sur les données DARPA'99

Nous avons choisi d'appliquer notre approche à ces données, car elles contiennent plusieurs attaques. Notre objectif est de tester l'utilité de notre approche, et comme nous avons déjà travaillé avec ces données pour la détection d'intrusions, nous avons certaines connaissances sur les attaques que contiennent ces données. Les données DARPA'99 [Darpa 1999] comportent cinq semaines de trafic réseau brut. Une présentation détaillée des données DARPA'99 est donnée dans le chapitre 7 (section 7.4.1). Pour nos expérimentations, nous utilisons les données de la semaine 2, 4 et 5 qui contiennent du trafic normal et anormal. Le tableau suivant montre quelques exemples d'alertes générées :

Type d'alertes	Message associé à l'alerte
TCP	WEB-MISC apache directory disclosure attempt
	WEB-IIS *.idc attempt
	MISC Source Port 20 to <1024
	ATTACK-RESPONSES directory listing
UDP	(spp_frag3) Short fragment, possible DoS attempt
	SHELLCODE x86 NOOP
	SNMP missing community string attempt
ICMP	ICMP Destination Unreachable Port Unreachable
	ICMP PING BSDtype
	ICMP Echo Reply

TAB. 8.6: Exemples d'alertes dans les données DARPA'99

Concernant ces données, nous supposons que les connaissances et préférences de l'administrateur

sont les suivantes :

Connaissances de l'administrateur réseau

- $\psi_1 = \forall x, \forall y\ SameIPsrc(x,y) \wedge SameIPdst(x,y) \wedge SamePortsrc(x,y) \wedge SamePortdst(x,y) \wedge SameSid(x,y) \wedge Timestamp\text{-}Smaller\text{-}than(x,y) \wedge Differ(x,y) \rightarrow Present\text{-}Alert(x) \wedge \neg Present\text{-}Alert(y)$.

Cette connaissance concerne la suppression de toutes les alertes redondantes.

Préférences de l'administrateur réseau

- $\psi_2 = \forall x,y\ Class(x, DoS) \wedge Class(y, other) \wedge Differ(x,y) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$
- $\phi_3 = \forall x,y\ Class(x, Probe) \wedge Class(y, other) \wedge Differ(x,y) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$
- $\phi_4 = \forall x,y\ Class(x, U2R) \wedge Class(y, other) \wedge Differ(x,y) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$
- $\phi_5 = \forall x,y\ Class(x, R2L) \wedge Class(y, other) \wedge Differ(x,y) \rightarrow Present\text{-}alert(x) \vec{\times} Present\text{-}alert(y)$

Ces quatre formules codent des préférences pour l'analyse des alertes DoS, R2L, U2R et Probe que Snort arrive à détecter. Les alertes qui ne sont pas détectées dans ces catégories, sont préférés en second degré (la classe Other). Les résultats concernant les données DARPA'99 sont affichés dans le tableau suivant :

	# d'alertes	# d'alertes préférées	Exemples d'attaques détectées
Données d'apprentissage (Semaine 2)	30204	2054 (23580 alertes répétitives)	ntinfoscan, guesstelnet, back portsweep, guessftp, snmpget
Données de test (Semaine 4)	19230	867 (11759 alertes répétitives)	ntinfoscan, guesstelnet, guessftp mailbomb, named, ftpwrite
Données de test (Semaine 5)	39212	1294 (28673 alertes répétitives)	named, guesstelnet, ntinfoscan sechole, yaga, casesen, portsweep
Total	**88646**	**4215** (64012 alerte répétitives)	

TAB. 8.7: Les alertes préférées des données DARPA'99

Nous remarquons que le taux de réduction des alertes est élevé, cela s'explique par deux différentes raisons. D'une part, Snort réussit dans la détection de plusieurs attaques et comme les signatures de la majorité des attaques détectées sont connues (1156 (DoS), 718(U2R), 553(R2L), 503(Probe) etc.), les signatures qui ne sont pas connues sont considérées comme des alertes à analyser en dernier. D'un autre coté, la connaissance concernant la suppression des alertes répétitives est très importante vu le nombre d'alertes qui permet de réduire. Si nous prenons le cas des alertes de la semaine 2, cette connaissance (suppression des alertes répétitives) a permet de supprimer 23580 alertes, ce qui présente 78.06%, c'est à dire que seulement 6624 alertes qui nécessitent l'analyse au lieu de 30204.

8.5 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle approche de corrélation d'alertes. Notre approche consiste essentiellement à prendre en considération des connaissances et préférences que l'administrateur réseau a sur le fonctionnement du système qu'il surveille, l'IDS qu'il utilise, les alertes, etc. Pour coder ces connaissances et préférences, notre approche se base sur la logique de représentation des préférences $\mathcal{FO}$-$\mathcal{MQCL}$ qui est une extension simple de la logique $\mathcal{MQCL}$ en un fragment de la logique du premier ordre. L'intérêt de cette approche réside dans la simplicité qu'elle offre pour exprimer des connaissances et des préférences de différentes structures (simples ou complexes). Rajoutant à cela, la possibilité d'ordonner les différentes alertes selon les connaissances et les préférences exprimées à l'aide de la définition des modèles préférées. Des alertes préférées sont présentées en premier à l'administrateur réseaux, si besoin des alertes préférées en second degré seront également présentées pour l'analyse, etc.

Notre approche est intéressante dans la mesure où un administrateur réseau qui surveille le système est le mieux placé pour savoir le type de failles du système que les attaques peuvent exploiter, le nombre et le type des alertes qui se produisent chaque jour, l'IDS qui produit le plus d'alertes dans le cas où plusieurs IDSs sont utilisés, etc. C'est pourquoi, cette approche donne la possibilité d'exprimer toutes ces informations et de les coder dans un cadre logique ayant une syntaxe est une sémantique très riche et très adaptée à ce type de situations.

Conclusion générale

Dans le cadre de cette thèse, nous avons traité trois problèmes importants : représentation des préférences dans un cadre logique permettant de raisonner avec des conditionnels, application des modèles graphiques pour le problème de la détection d'intrusions, notamment en présence d'informations incertaines et application de la logique des préférences au problème de la corrélation d'alertes. Les différents résultats obtenus affirment que les techniques d'intelligence artificielle que nous avons utilisées (modèles graphiques, raisonnement sous incertitude et en présence de préférences) peuvent être adaptées de manière satisfaisante à d'autres problèmes plus sensibles et plus préoccupants comme la détection d'intrusions et la corrélation d'alertes.

La première problématique à laquelle nous nous sommes intéressés concerne la représentation et modélisation des préférences. Nous sommes intéressés, dans un premier temps, à étudier différentes approches de représentation des préférences dans un cadre général. Par la suite, nous nous sommes focalisés particulièrement sur l'étude de la nouvelle logique du choix qualitatif ($\mathcal{QCL}$). Cette logique, qui est une extension de la logique propositionnelle classique, a montré son utilité en représentation des préférences par ses fortes propriétés, en particulier la disjonction ordonnée, l'optionalité et le degré associé à la satisfaction des préférences d'un agent. Cependant, elle présente quelques problèmes, ce qui rend son champ d'utilisation limité, notamment en présence d'un ensemble de préférences accompagnées par une négation et le raisonnement sur les règles conditionnelles de la forme *"Si A est préféré à B alors C est préféré à D"*. Compte tenu de ces problèmes, notre objectif a été de proposer des solutions afin de remédier aux limites de la logique $\mathcal{QCL}$.

Ainsi, après avoir cerné les limites de cette logique, nous avons proposé quelques révisions et modifications pour surmonter ces limites. La première modification importante concerne une nouvelle définition de la négation des préférences. La sémantique de notre nouvelle négation est différente de celle de la logique $\mathcal{QCL}$, dans le sens où la satisfaction des préférences des agents exprimées par la négation est plus satisfaisante et traitée comme d'autres préférences simples ou générales. Dans le cadre de $\mathcal{QCL}$, une préférence représentée par une formule de choix générale de la forme $\neg(A\vec{\times}B)$ est équivalente à la formule propositionnelle $(\neg A \wedge \neg B)$ qui considère que les alternatives A et B sont exclues, c'est-à-dire que la seule possibilité de satisfaire cette préférence $(\neg(A\vec{\times}B))$ est lorsque "A et B sont fausses". Alors que dans le cas de notre définition, trois possibilités de satisfaire la préférence avec deux degrés différents : lorsque *"A et B sont fausses"*, *"A est fausse, B est vraie"*, cette préférence est satisfaite avec le degré 1 et lorsque *"A est vraie, B est fausse"*, elle est satisfaite avec le degré 2. L'autre avantage de la nouvelle négation concerne le raisonnement avec des règles de la forme *"Si A est préféré à B alors C est préféré à D"* qui est plus satisfaisant et ne pose aucun problème en utilisant la nouvelle relation d'inférence.
La première modification de la logique $\mathcal{QCL}$ au niveau de la négation a donné une nouvelle logique que nous avons appelée *Logique du Choix Qualitatif Minimale* ($\mathcal{MQCL}$) qui se base sur

le même langage que $\mathcal{QCL}$ (connecteurs et vocabulaire) et sa relation d'inférence se base sur la fonction de normalisation $\mathcal{N}_{\mathcal{MQCL}}$ permettant de transformer d'une manière équivalente tout ensemble de préférences générales et complexes en un ensemble de préférences simples.

La nouvelle logique $\mathcal{MQCL}$ est intéressante du fait que, d'une part, sa relation d'inférence est satisfaisante et d'autre part s'avère largement suffisante pour modéliser certains problèmes comme la corrélation d'alertes. Cependant, elle n'est pas très adaptée au raisonnement avec d'autres types de préférences telles que les préférences prioritaires et positives. Par conséquent, d'autres modifications à la logique $\mathcal{QCL}$ sont nécessaires afin de pouvoir raisonner sur d'autres types de préférences. Ainsi, nous avons proposé de nouvelles modifications au niveau de la conjonction et disjonction des préférences. Nous avons donné deux nouvelles définitions de la conjonction (resp. disjonction) : la première définition (conjonction et disjonction) permet de représenter des préférences ayant différents niveaux de priorité et la seconde est adaptée à représenter des préférences positives. Ces nouvelles modifications ont donné le résultat de deux autres logiques : la première est appelée *Logique du Choix Qualitatif Prioritaire* ($\mathcal{PQCL}$) et la seconde est appelée *Logique du Choix Qualitatif Positive* ($\mathcal{QCL}+$). Chacune de ces logiques permet de surmonter les limites de $\mathcal{QCL}$ et propose de normaliser les préférences générales par l'utilisation de fonctions de normalisations ($\mathcal{N}_{\mathcal{PQCL}}$ pour la logique $\mathcal{PQCL}$ et $\mathcal{N}_{\mathcal{QCL}+}$ pour $\mathcal{QCL}+$).

Pour les applications à la détection d'intrusions et à la corrélation d'alertes, nous avons considéré que ces problèmes sont avant tout des problèmes de raisonnement sur la nature des informations que contiennent les données. C'est pourquoi, en plus de représentation et modélisation des préférences, nous nous sommes confrontés à étudier plusieurs autres points : définition et extraction d'attributs, formatage du trafic réseau brut, modèles graphiques probabilistes, algorithmes d'apprentissage automatique/classification.

Concernant le problème de la détection d'intrusions, nous nous sommes intéressés au fait que le problème se base dans un premier temps à définir des attributs. Ces attributs servent à décrire les différents événements normaux et anormaux contenus dans un trafic réseau et aussi pour pouvoir modéliser le problème par un modèle graphique. Le jeu d'attributs que nous avons défini englobe des attributs de base (durée des événements, service et protocole utilisés, etc.), attributs de contenus (nombre d'accès en mode root, nombre de login échoués, etc.), attributs relatifs au trafic réseau calculés sur un intervalle de quelques secondes (resp. 2, 5, 6, etc.), attributs relatifs à un hôte de destination particulier pendant une plage de connexions (100 dernières connexions, 200, etc.) et attributs orientés protocoles (flags incorrects, trafic anormal, etc.). En tout, nous avons défini 44 attributs. Pour extraire ces attributs à partir d'un trafic réseau brut, nous avons développé un outil de formatage offrant plusieurs fonctionnalités telles que l'extraction d'attributs, la construction de connexions, les statistiques sur le trafic, etc. Les attributs peuvent être extraits de deux manières différentes :

- Soit attendre la fin d'une connexion pour extraire ses attributs. Dans ce cas, les données décrivant cette connexion sont complètes.
- Soit extraire ces attributs avant la fin d'une connexion. Dans ce cas, les données décrivant cette connexion sont incomplètes.

Nous avons proposé deux types de formatage : formatage hors ligne et formatage en temps réel. Lorsque les attributs sont extraits à partir du trafic réseau enregistré dans des fichiers, il s'agit du formatage hors ligne et lorsqu'ils sont extraits sur un trafic en cours de capture, on parle du formatage en temps réel.

Notre objectif en définissant des attributs et les extraire à l'état fini ou non fini est, d'une part, tester la pertinence des attributs définis et d'autre part détecter les attaques le plus tôt possible. Pour nos expérimentations, nous avons utilisé les réseaux bayésiens qui sont des modèles graphiques probabilistes largement appliqués dans le domaine de la détection d'intrusions. Les résultats expérimentaux basés sur les attributs définis sont satisfaisants et confirment leur pertinence. Quant à la détection avec des connexions non finies ou incomplètes (détection en temps réel), les résultats obtenus sont également satisfaisants.

Concernant la corrélation d'alertes, qui a pour objectif de réduire le nombre d'alertes générées par les systèmes de détection d'intrusions, nous nous sommes basés sur le fait qu'un administrateur réseau est le mieux placé pour savoir le fonctionnement de son système, le type de failles qui existent, le nombre et le type d'alertes générées, les alertes qui nécessitent l'analyse ou non, etc. Ainsi, toutes ces informations peuvent être représentées par deux éléments :

- Toute information que l'administrateur réseau peut obtenir par expérience, ou qu'il peut avoir sur les alertes, est considérée comme une connaissance ou une contrainte d'intégrité.
- Toute information pouvant aider l'administrateur à écarter ou ignorer certaines alertes et favoriser d'autres est considérée comme une préférence.

Notre intérêt ici, s'est focalisé sur le raisonnement en présence des préférences. Ainsi, nous avons proposé une nouvelle approche basée sur l'une des logiques que nous avons développées. Il s'agit de la logique du choix qualitatif minimale $\mathcal{MQCL}$. Comme le langage de cette logique est propositionnel, nous avons proposé dans un premier temps d'étendre ce langage en un fragment de la logique du premier ordre qui est un cadre plus général. Les connaissances de l'administrateur réseau sont représentées par des formules propositionnelles universellement quantifiées, ses préférences simples sont représentées par des formules de choix de base universellement quantifiées et ses préférences générales ou plus complexes sont représentées par des formules de choix générales universellement quantifiées. Pour trouver les alertes préférées qui sont les alertes qui satisfont les connaissances et les préférences de l'administrateur réseau, notre approche se base sur la définition des modèles préférés dans le cadre de la logique $\mathcal{MQCL}$. Le principe de fonctionnement de notre modèle est le suivant :

- Représenter toutes les connaissances de l'administrateur réseau dans une base de connaissances, notée K_2.
- Représenter toutes ses préférences dans une base de préférences, notée T.
- Coder les connaissances de la base K_2 et les préférences de T dans le cadre de la logique $\mathcal{MQCL}$.
- Extraire une base de faits qui représente des informations sur chaque alerte à analyser.
- Appliquer la relation d'inférence de la logique $\mathcal{MQCL}$ en tenant compte des trois bases (faits, connaissances et préférences).
- Appliquer la définition des modèles préférés pour ne présenter à l'administrateur que les alertes préférées. Si besoin, des alertes préférées en second degré peuvent être également présentées, etc.

Nous avons validé notre approche sur deux bases de données (DARPA'99 et DADDi) et les résultats sont satisfaisants même si les règles que nous avons choisies ne sont pas données par un expert du domaine (mais on s'est inspiré des informations obtenues lors des discussions échangées dans le cadre du projet DADDi). En particulier, cette approche est intéressante si nous disposons

de vraies règles expertes.

Ce travail ouvre la voie pour d'autres points à explorer. Nous pensons en particulier à la détection en temps réel dans un cadre plus général. Il est, par exemple, important de définir deux profils (normal et anormal) à partir des connexions incomplètes, c'est-à-dire que le profil d'apprentissage sera basé sur des données incomplètes. En effet, lorsque les connexions sont complètes, les caractéristiques des attaques sont différentes de celles des connexions incomplètes. Un autre point important concerne la détection des moments d'apparition des signatures d'attaques (dans nos résultats expérimentaux, nous avons constaté qu'il existe des attaques qui peuvent être détectées dès les premiers paquets, alors que d'autres sont détectées après un certain nombre de paquets). D'autre part, nous nous intéresserons également à mettre l'accent sur l'application de la règle de Jeffrey et Kinematics [Chan & Darwiche 2003] pour la révision des données incomplètes. L'idée principale se résume sur l'exemple suivant : Le flag d'une connexion UDP peut prendre une des valeurs suivantes (SF, S0, S1, OTH). Supposons que nous nous intéressons à savoir la valeur du flag à un instant donné sachant qu'à cet instant, nous avons une observation concernant le nombre de bytes source (Src_bytes) et une autre concernant le nombre de bytes de destination (Dst_bytes). Selon Jeffrey [Chan & Darwiche 2003], si nous considérons une distribution de probabilités initiale, notée Pr sur l'état du flag, Pr' est la nouvelle distribution de probabilités sachant l'observation concernant Src_bytes et Pr'' est une autre distribution de probabilités sachant Dst_bytes, alors ces trois distributions représentent des relations entre elles. C'est pourquoi, nous nous intéressons à exploiter ce type de liens pour connaître le plus tôt possible la valeur du flag.

Bibliographie

[Allais 1953] Maurice Allais. « Le comportement de l'homme rationnel devant le risque : Critique des postulats et axiomes de l'école américaine ». *Jornal of Econometrica*, 21 :503–546, 1953.

[Amgoud & Dupin de Saint Cyr 2008] Leila Amgoud et Florence Dupin de Saint Cyr. « Measures for persuasion dialogs : A preliminary investigation ». Dans Philippe Besnard, Sylvie Doutre, et Anthony Hunter, éditeurs, *Computational models of argument (COMMA)*, pages 13–24, Toulouse, France, mai 2008. IOS Press.

[Anderson 1980] James P. Anderson. *Computer security threat monitoring and surveillance*. Anderson Company, Fort Washington, Pennsylvania, 1980.

[Armstrong 1939] W.E. Armstrong. « The determinateness of the utility function ». *The Economic Journal*, 49 :453–467, 1939.

[Armstrong 1948] W.E. Armstrong. « Uncertainty and utility function ». *Economics Journal*, 58 :1–10, 1948.

[Barbará *et al.* 2001] Daniel Barbará, Julia Couto, Sushil Jajodia, et Ningning Wu. « ADAM : a testbed for exploring the use of data mining in intrusion detection ». *Journall of ACM SIGMOD Record*, 30(4) :15–24, 2001.

[Barse & Jonsson 2004] Emilie Lundin Barse et Erland Jonsson. « Extracting Attack Manifestations to Determine Log Data Requirements for Intrusion Detection ». Dans *Annual Computer Security Applications Conference (ACSAC'04)*, pages 158–167, 2004.

[Ben Amor *et al.* 2004] Nahla Ben Amor, Salem Benferhat, et Zied Elouedi. « Réseaux Bayésiens Naifs et Arbres de Décision dans les Systèmes de Détection d'Intrusions ». Dans *quatorzième Congrés Francophone AFRIF AFIA sur la Reconnaissance des Formes et l'Intelligence Artificielle (RFIA'04)*, pages 1175–1184, Toulouse France, jan 2004.

[Benferhat & Kaci 2003] Salem Benferhat et Souhila Kaci. « Logical representation and fusion of prioritized information based on guaranteed possibility measures : application to the distance-based merging of classical bases ». *Journal of Artificial Intelligence*, 148(1-2) :291–333, 2003.

[Benferhat & Sedki 2007] Salem Benferhat et Karima Sedki. « A Revised Qualitative Choice Logic for Handling Prioritized Preferences ». Dans *Ninth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty(ECSQARU'07)*, pages 635–647, oct 2007.

[Benferhat & Sedki 2008a] Salem Benferhat et Karima Sedki. « Alert Correlation based on a Logical Handling of Administrator Preferences and Knowledge ». Dans

International Conference on Security and Cryptography(SECRYPT'08), pages 50–56, Porto, Portugal, jul 2008.

[Benferhat & Sedki 2008b] Salem Benferhat et Karima Sedki. « Two alternatives for handling preferences in qualitative choice logic ». *Fuzzy Sets and Systems Journal (FSS'08)*, 159(15) :1889–1912, august 2008.

[Benferhat & Smaoui 2007] Salem Benferhat et Salma Smaoui. « Hybrid possibilistic networks ». *International Journal of Approximate Reasoning*, 44(3) :224–243, mar 2007.

[Benferhat & Tabia 2005] Salem Benferhat et Karim Tabia. « On the combination of naive Bayes and decision trees for intrusion detection ». Dans *CIMCA'05 : Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation*, pages 211–216, Washington, DC, USA, 2005. IEEE Computer Society.

[Benferhat & Tabia 2008a] Salem Benferhat et Karim Tabia. « Classification features for detecting server-side and client-side web attacks ». Dans *23rd International Security Conference(SEC'08)*, pages 729–733. Springer LNCS, 2008.

[Benferhat & Tabia 2008b] Salem Benferhat et Karim Tabia. « Novel and anomalous behavior detection using Bayesian network classifiers ». Dans *International Conference on Security and Cryptography(SECRYPT'08)*, pages 13–20, 2008.

[Benferhat *et al.* 2001] Salem Benferhat, Didier Dubois, et Henri Prade. « Towards a Possibilistic Logic Handling of Preferences ». *Journal of Applied Intelligence*, 14(3) :303–317, 2001.

[Benferhat *et al.* 2002] Salem Benferhat, Souhila Kaci, Dédier Dubois, et Henri Prade. « Bipolar representation and fusion of preferences in the possibilistic logic framework ». Dans *Proceedings of 8 th International Conference on Principles of Knowledge Representation and Reasoning KR'2002*, pages 421–432, Toulouse, april 2002. Morgan Kaufmann Publishers, Inc.

[Benferhat *et al.* 2004] Salem Benferhat, Gérard Brewka, et Daniel Le Berre. « On the Relationship Between Qualitative Choice Logic and Possibilistic Logic ». Dans *Tenth International Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems(IPMU'04)*, volume 2, pages 951–957, Perugia, Italy, jul 2004.

[Benferhat *et al.* 2006] Salem Benferhat, Daniel Le Berre, et Karima Sedki. « An Alternative Inference for Qualitative Choice Logic ». Dans Anna Perini Paolo Traverso Gerhard Brewka, Silvia Coradeschi, éditeur, *17th European Conference on Artificial Intelligence(ECAI'06)*, pages 741–742, Riva del Garda, Italie, 2006. IOS Press.

[Benferhat *et al.* 2007] Salem Benferhat, Karima Sedki, et Karim Tabia. « Reprocessing Rough Network Traffic for Intrusion Detection Purposes ». Dans *IADIS : International Conference Telecommunications, Networks and Systems*, Lisbon, Portugal, 2007.

[Benferhat *et al.* 2008] Salem Benferhat, Tayeb Kenaza, et Aïcha Mokhtari. « Tree-Augmented Naïve Bayes for Alert Correlation ». Dans *3rd conference on Advances in Computer Security and Forensics (ACSF'08)*, pages 45–52, jul 2008.

[Bishop 1995] Matt Bishop. « A Standard Audit Trail Format ». Dans *Proceedings of 18th NIST-NCSC National Information Systems Security Conference*, pages 136–145, 1995.

[Bistarelli *et al.* 2005] Stefano Bistarelli, Maria Silvia Pini, Francesca Rossi, et Kristen Brent Venable. « Positive and negative preferences ». Dans *Proceedings of the 7th International Workshop on Preferences and Soft Constraints (SOFT'05)*, oct 2005.

[Bonzon 2007] Elise Bonzon. « *Modélisation des interactions entre agents rationnels : les jeux booléens* ». PhD thesis, Université Paul Sabatier, 2007.

[Borgelt *et al.* 2002] Christian Borgelt, Jorg Gebhardt, et Rudolf Kruse. « Graphical Models ». Dans *Proceedings of International School for the Synthesis of Expert Knowledge (ISSEK'98)*, pages 51–68. Wiley, 2002.

[Boutilier 1994] Craig Boutilier. « Toward a Logic for Qualitative Decision Theory ». Dans *KR'94*, pages 75–86, 1994.

[Boutilier *et al.* 1999] Craig Boutilier, Ronen I. Brafman, Holger H. Hoos, et David Poole. « Reasoning with Conditional Ceteris Paribus Preference Statements ». Dans *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99)*, pages 71–80, 1999.

[Boutilier *et al.* 2004] Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, et David Poole. « CP-Nets : A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements ». *Journal of Artificial Intelligence Research (JAIR)*, 21, 2004.

[Bouyssou & Vincke 2003] Denis Bouyssou et Philippe Vincke. « Relations binaires et modélisation des préférences ». Rapport Technique TR/SMG/2003-02, SMG, Université Libre de Bruxelles, 2003.

[Bouzida 2006] Yacine Bouzida. « *Application de l'analyse en composante principale pour la détection d'intrusion et détection de nouvelles attaques par apprentissage supervisé* ». PhD thesis, RSM - Dépt. Réseaux, Sécurité et Multimédia (Institut TELECOM/TELECOM Bretagne), UR1 - Université de Rennes 1, 2006.

[Brewka 1989] Gerhard Brewka. « Preferred Subtheories : An Extended Logical Framework for Default Reasoning ». Dans *IJCAI'89*, pages 1043–1048, 1989.

[Brewka 1994] Gerhard Brewka. « Reasoning about Priorities in Default Logic ». Dans *AAAI*, pages 940–945, 1994.

[Brewka 2002] Gerhard Brewka. « Logic programming with ordered disjunction ». Dans *Eighteenth national conference on Artificial intelligence*, pages 100–105, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[Brewka *et al.* 2004] Gerard Brewka, Salem Benferhat, et Daniel Le Berre. « Qualitative Choice Logic ». *Artificial Intelligence Journal(AIJ)*, 157(1-2) :203–237, 2004.

[Carvalho & Oliveira 2007] Alexandra M. Carvalho et Arlindo L. Oliveira. « Learning Bayesian Networks Consistent with the Optimal Branching ». Dans *Proceedings of the Sixth International Conference on Machine Learning and*

Applications (ICMLA), pages 369–374, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[Chan & Darwiche 2003] Hei Chan et Adnan Darwiche. « On the Revision of Probabilistic Beliefs Using Uncertain Evidence ». Dans *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 99–105, San Francisco, California, 2003. Morgan Kaufmann Publishers.

[Chein & laure Mugnier 1992] Michel Chein et Marie laure Mugnier. « Conceptual graphs : Fundamental notions ». *Revue d'Intelligence Artificielle*, 6 :365–406, 1992.

[Cheng & Greiner 2001] Jie Cheng et Russell Greiner. « Learning Bayesian Belief Network Classifiers : Algorithms and System ». Dans *Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence : Advances in Artificial Intelligence*, pages 141–151, 2001.

[Chickering 1996] David Maxwell Chickering. « Learning Bayesian networks is NP-complete ». Dans *Learning from Data : Artificial Intelligence and Statistics V*, pages 121–130. Springer-Verlag, 1996.

[Chimphlee *et al.* 2006] Witcha Chimphlee, Abdul Hanan Abdullah, Mohd Noor Md Sap, Surat Srinoy, et Siriporn Chimphlee. « Anomaly-Based Intrusion Detection using Fuzzy Rough Clustering ». Dans *ICHIT '06 : Proceedings of the 2006 International Conference on Hybrid Information Technology*, pages 329–334, Washington, DC, USA, 2006. IEEE Computer Society.

[Chisholm & Sosa 1966] Roderick M. Chisholm et Ernest Sosa. « On the logic of Intrinsically Better ». *American Philosophical Quarterly*, 3 :244–249, 1966.

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

[Chow & Liu 1968] C. Chow et C. Liu. « Approximating discrete probability distributions with dependence trees ». *Information Theory, IEEE Transactions on*, 14(3) :462–467, 1968.

[CISL 2000] CISL. « Common Intrusion Detection Framework Working Group. A CISL Tutorial ». Dans *http//www.gidos.org/tutorial.html*, 2000.

[Coombs & Smith 1973] C.H. Coombs et J.E.K Smith. « On the detection of structures in attitudes and developmental processes ». *Psychological Reviews*, 80 :337–351, 1973.

[Cooper & Herskovits 1992] Gregory F. Cooper et Edward Herskovits. « A Bayesian Method for the Induction of Probabilistic Networks from Data ». *Mach. Learn.*, 9(4) :309–347, 1992.

[Coste-Marquis & Marquis 2004] Sylvie Coste-Marquis et Pierre Marquis. « On stratified belief base compilation ». *Annals of Mathematics and Artificial Intelligence(AMAI)*, 42(4) :399–442, décembre 2004.

[Crosbie *et al.* 1996] Mark Crosbie, Bryn Dole, Todd Ellis, Ivan Krsul, et Eugene Spafford. « IDIOT - Users Guide ». Rapport Technique, 1996.

[Cuppens & Autrel 2005] Frédéric Cuppens et Fabien Autrel. « Using an intrusion detection alert similarity operator to aggregate and fuse alerts ». Dans *In the 4th Conference on Security and Network Architectures SAR'2005*, pages 6–10, Batz sur Mer (France), 2005.

[Cuppens & Miège 2002] Frédéric Cuppens et Alexandre Miège. « Alert Correlation in a Cooperative Intrusion Detection Framework ». Dans *SP'02 : Proceedings of the 2002 IEEE Symposium on Security and Privacy*, page 202, Washington, DC, USA, 2002. IEEE Computer Society.

[Cuppens & Ortalo 2000] Frédéric Cuppens et Rodolphe Ortalo. « LAMBDA : A Language to Model a Database for Detection of Attacks ». Dans *RAID'00 : Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection*, pages 197–216, London, UK, 2000. Springer-Verlag.

[Cuppens 2001] Frédéric Cuppens. « Managing Alerts in a Multi-Intrusion Detection Environment ». Dans *Annual Computer Security Applications Conference (ACSAC'01)*, page 0022, USA, 2001.

[Curry & Debar 2001] D. Curry et Hervé Debar. « Intrusion Detection Message Exchange Format Data Model and Extensible Markup Language (XML) Document Type Definition, draft-itetfidwg- idmef-xml-03.txt ». 2001.

[Dain & Cunningham 2001] Oliver Dain et Robert K. Cunningham. « Fusing a heterogeneous alert stream into scenarios ». Dans *Proceedings of the 2001 ACM workshop on Data Mining for Security Applications*, pages 1–13, 2001.

[Darpa 1999] Darpa. « MIT Lincoln Laboratory, DARPA Intrusion Detection Evaluation Data Sets, http ://www.ll.mit.edu/IST/ideval/data/data_index.html ». 1999.

[Darwiche & Marquis 2002] Adnan Darwiche et Pierre Marquis. « A knowledge compilation map ». *Journal of Artificial Intelligence Research (JAIR)*, 17 :229–264, 2002.

[Debar & Wespi 2001] Hervé Debar et Andreas Wespi. « Aggregation and correlation of intrusion-detection alerts ». Dans *Recent Advances in Intrusion Detection (RAID'01)*, pages 85–103, 2001.

[Debreu 1954] Gérard Debreu. Representation of a preference ordering by a numerical function. Dans R. Thrall, C.H. Coombs, et R. Davies, éditeurs, *Decision Processes*, pages 159–175. John Wiley & Sons, New York, 1954.

[Denning 1987] Dorothy E. Denning. « An Intrusion-Detection Model ». *IEEE Trans. Softw. Eng.*, 13(2) :222–232, 1987.

[Deraison 2000] Renaud Deraison. « The Nessus Attack Scripting Language Reference Guide ». Dans *http//www.nissus.org*, 2000.

[Domshlak 2002] Carmel Domshlak. « *Modelling and reasoning about preferences with CP-nets* ». thesis submitted in partial fulfillment of the requirements for the degree of doctor of philosophy, University of the Negev, jul 2002.

[Dowell & Ramstedt 1990] C. Dowell et P. Ramstedt. « The computerwatch date reduction tool ». Dans *13 National Computer security Conference*, Washington, DC, 1990.

[Doyle *et al.* 1991] Jon Doyle, Yoav Shoham, et Michael P. Wellman. « A Logic of Relative Desire (Preliminary Report) ». Dans *ISMIS'91 : Proceedings of the 6th International Symposium on Methodologies for Intelligent Systems*, pages 16–31, London, UK, 1991. Springer-Verlag.

[Dubois & Prade 1995] Didier Dubois et Henri Prade. « Possibility theory as a basis for qualitative decision theory ». Dans *Proceedings of the 14th International Joint Conference on Artificial Intelligence, IJCAI'95*, pages 1924–1930. Morgan Kaufmann, San Francisco, 1995.

[Dubois & Prade 2002] Didier Dubois et Henri Prade. « Bipolarity in Possibilistic Logic and Fuzzy Rules ». Dans *SOFSEM'02 : Proceedings of the 29th Conference on Current Trends in Theory and Practice of Informatics*, pages 168–173, London, UK, 2002. Springer-Verlag.

[Dubois *et al.* 1994] Didier Dubois, Jérôme Lang, et Henri Prade. « Automated reasoning using possibilistic logic : semantics, belief revision and variable certainty weights ». *IEEE Trans. on Data and Knowledge Engineering*, 6(1) :64–71, 1994.

[Dubois *et al.* 1998] Didier Dubois, Lluis Godo, Henri Prade, et Adriana Zapico. « Making decision in a qualitative setting : from decision under uncertainty to case-based decision ». Dans *6th Int. Conf. Principles of Knowledge Representation and Reasoning , Trento, Italy,* , San Francisco, CA, juin 1998. Morgan Kaufmann.

[Dubois *et al.* 2000] Didier Dubois, Petr Hájek, et Henri Prade. « Knowledge-Driven versus data-driven logics ». *Journal of Logic, Language, and Information*, 9(1) :65–89, 2000.

[Dubois *et al.* 2002] Didier Dubois, Hélène Fargier, Henri Prade, et Patrice Perny. « Qualitative decision theory : from savage's axioms to nonmonotonic reasoning ». *Journal of the ACM (JACM)*, 49(4), jul 2002.

[Dubois *et al.* 2004] Dédier Dubois, Souhila Kaci, et Henré Prade. « Bipolarity in reasoning and decision – An introduction. The case of the possibility theory framework ». Dans *Proceedings of Information Processing and Management of Uncertainty in Knowledge-Based Systems Conference (IPMU'04)*, pages 959–966, Perugia, Italy, jul 2004.

[Dubois *et al.* 2006] Didier Dubois, Souhila KACI, et Henri Prade. « Approximation of Conditional Preferences Networks ». Dans *IEEE International Conference on Fuzzy Systems(FUZZ-IEEE'06)*, Vancouver, Canada, 2006. 8 pages (CD-ROM).

[Eckmann *et al.* 2002] Steven T. Eckmann, Giovanni Vigna, et Richard A. Kemmerer. « STATL : an attack language for state-based intrusion detection ». *Journal of Computer Security*, 10(1-2) :71–103, 2002.

[Ehrgott 2000] Matthias Ehrgott. *Multicriteria Optimization*, volume 491 de *Lecture Notes in Economics and Mathematical Systems*. Springer Verlag, Berlin, 2000.

[Elkan 2000] Charles Elkan. « Results of the KDD'99 classifier learning ». *ACM SIGKDD Explorations Newsletter*, 1(2), January 2000.

[Fargier & Sabbadin 2005] Hélène Fargier et Régis Sabbadin. « Qualitative decision under uncertainty : back to expected utility ». *Journal of Artificial Intelligence*, 164 :245–280, 2005.

[Fargier *et al.* 1993] Hélène Fargier, Jérôme Lang, et Thomas Schiex. « Selecting preferred solutions in fuzzy constraint satisfaction problems ». Dans *In*

Proceedings of the first European Congress on Fuzzy and Intelligent Technologies (EUFIT'93), pages 1128–1134, Aachen, Germany, 1993.

[Fishburn 1970] Peter C. Fishburn. *Utility Theory for Decision-Making*. John Wiley & Sons, New York, 1970.

[Fishburn 1999] Peter C. Fishburn. « Preference structures and their numerical presentations ». *Theoretical Computer Science*, 217 :359–389, 1999.

[François & Leray 2004] Olivier François et Philippe Leray. « Etude Comparative d'Algorithmes d'Apprentissage de Structure dans les Réseaux Bayésiens ». *Journal électronique d'intelligence artificielle (JEDAI)*, 5(39) :1–19, 2004.

[Friedman & Goldszmidt 1996] Nir Friedman et Moises Goldszmidt. « Building Classifiers Using Bayesian Networks ». Dans *Proceedings of the thirteenth national conference on artificial intelligence*, pages 1277–1284. AAAI Press, 1996.

[Friedman *et al.* 1997] Nir Friedman, Dan Geiger, et Moises Goldszmidt. « Bayesian Network Classifiers ». *Machine Learning*, 29(2-3) :131–163, 1997.

[Fung & Favero 1994] Robert Fung et Del B. Favero. « Backward Simulation in Bayesian Networks ». Dans *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 227–234, San Francisco, CA, USA, july 1994.

[Ghirardato *et al.* 2004] Paolo Ghirardato, Fabio Maccheroni, et Massimo Marinacci. « Differentiating ambiguity and ambiguity attitude ». *Journal of Economic Theory*, 118(2) :133–173, October 2004.

[Habra *et al.* 1992] Naji Habra, Baudouin Le Charlier, Abdelaziz Mounji, et Isabelle Mathieu. « ASAX : Software Architecture and Rule-Based Language for Universal Audit Trail Analysis ». Dans *ESORICS'92*, pages 435–450, London, UK, 1992. Springer-Verlag.

[Halldén 1957] Sören Halldén. *On the Logic of Better*. Library of Theoria, Lund, 1957.

[Hansson 1996] Sven Ove Hansson. « What is ceteris paribus preference ? ». *Journal of Philosophical Logic*, 425(3) :307–332, 1996.

[Heckerman 1999] David Heckerman. « A tutorial on learning with Bayesian networks ». pages 301–354, 1999.

[Hertz *et al.* 1991] John Hertz, Anders Krogh, et Richard G. Palmer. *Introduction to the theory of neural computation*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1991.

[Itzhak & David 1989] Gilboa Itzhak et Schmeidler David. « Maxmin expected utility with non-unique prior ». *Journal of Mathematical Economics*, 18(2) :141–153, April 1989.

[Jacobson *et al.* 2000] Van Jacobson, Craig Leres, et Steve McCanne. « Tcpdump 3.5 documentation ». Dans *http ://www.tcpdump.org*, 2000.

[Jaeger 2004] Manfred Jaeger. « Probabilistic decision graphs-combining verification and AI techniques for probabilistic inference ». *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 12 :19–42, 2004.

[Jakobson & Weissman 1993] Gabriel Jakobson et Mark Weissman. « Alarm correlation ». *IEEE Network Journal*, 7(6) :52–59, 1993.

[Jensen 1996] Finn V. Jensen. *An Introduction to Bayesian Networks*. UCL Press., 1996.

[Jensen 2001] Finn V. Jensen. *Bayesian Networks and Decision Graphs*. Information Science and Statistics. Springer, July 2001.

[Julisch 2003] Klaus Julisch. « Clustering Intrusion Detection Alarms to Support Root Cause Analysis ». *Journal ACM Transactions on Information and System Security*, 6 :443–471, 2003.

[Kahneman & Tversky 1979] Daniel Kahneman et Amos Tversky. « Prospect theory : An analysis of decision under risk ». *Journal of Econometrica*, 47 :263–291, 1979.

[KDD 1999] KDD. « http ://kdd.ccs.uci.edu/databases/kddcup99 ». 1999.

[Keeney & Raiffa 1976] Ralph L. Keeney et Howard Raiffa. *Decisions with multiple objectives : Preferences and value tradeoffs*. J. Wiley, New York, 1976.

[Keogh & Pazzani 2002] Eamonn J. Keogh et Michael J. Pazzani. « Learning the Structure of Augmented Bayesian Classifiers ». *International Journal on Artificial Intelligence Tools*, 11(4) :587–601, 2002.

[Klibanoff *et al.* 2005] Peter Klibanoff, Massimo Marinacci, et Sujoy Mukerji. « A Smooth Model of Decision Making under Ambiguity ». *Journal of Econometrica*, 73(6) :1849–1892, November 2005.

[Koziol 2003] Jack Koziol. *Intrusion Detection with Snort*. Sams, Indianapolis, IN, USA, 2003.

[Kruegel *et al.* 2003] Christopher Kruegel, Darren Mutz, William Robertson, et Fredrik Valeur. « Bayesian event classification for intrusion detection ». Dans *19th Annual Computer Security Applications Conference, LasVegas*. IEEE Computer Society, December 2003.

[Kumar & Spafford 1995] Sandeep Kumar et Eugene H. Spafford. « A software architecture to support misuse intrusion detection ». Dans *Proceedings of the 18th National Information Security Conference*, pages 194–204, 1995.

[Lang 1996] Jérôme Lang. « Conditional Desires and Utilities : an Alternative Logical Approach to Qualitative Decision Theory ». Dans *ECAI*, pages 318–322, 1996.

[Lang 2003] Jérôme Lang. « *Contribution à l'étude de modèles, de langages et d'algorithmes pour le raisonnement et la prise de décision en intelligence artificielle* ». Habilitation à diriger des recherches, Université Paul Sabatier, Toulouse, France, septembre 2003.

[Lang *et al.* 2002] Jérôme Lang, Leendert Van Der Torre, et Emil Weydert. « Utilitarian Desires ». *Dans Autonomous Agents and Multi-Agent Systems*, 5(3) :329–363, 2002.

[Lauritzen & Spiegelhalter 1988] Steffen L. Lauritzen et David J. Spiegelhalter. « Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems ». *Journal of the Royal Statistical Society, Series B*, 50 :157–224, 1988.

[Lee & Xiang 2001] Wenke Lee et Dong Xiang. « Information-theoretic measures for anomaly detection ». Dans *Proceedings of the 2001 IEEE Symposium on Security and Privacy*, pages 130–143, 2001.

[Lee 1999] Wenke Lee. « *A Data Mining Framework for Constructing Features and Models for Intrusion Detection Systems* ». PhD thesis, Computer Science Department, Columbia University, New York, june 1999.

[Lee *et al.* 1998] Wenke Lee, Salvatore J. Stolfo, et Kui W. Mok. « Mining Audit Data to Build Intrusion Detection Models ». Dans *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 66–72. AAAI Press, 1998.

[Lee *et al.* 1999] Wenke Lee, Salvatore J. Stolfo, et Kui W. Mok. « Mining in a data-flow environment : experience in network intrusion detection ». Dans *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99)*, pages 114–124, 1999.

[Lee *et al.* 2000] Wenke Lee, Salvatore J. Stolfo, et Kui W. Mok. « Adaptive Intrusion Detection : A Data Mining Approach ». *Artificial Intelligence Review*, 14 :533–567, 2000.

[Leray 2006] Philippe Leray. « *Réseaux bayésiens : apprentissage et modélisation de systèmes complexes* ». Mémoire présenté en vue de l'obtention de l'habilitation à diriger des recherches, département ASI, INSA, Rouen, 2006.

[Lindqvist 1999] Ulf Lindqvist. « Detecting computer and network misuse through the production-based expert system toolset (P-BEST) ». Dans *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, pages 146–161, 1999.

[Lippmann *et al.* 2000] Richard Lippmann, Joshua W. Haines, David J. Fried, Jonathan Korba, et Kumar Das. « Analysis and Results of the 1999 DARPA Off-Line Intrusion Detection Evaluation ». Dans *RAID '00 : Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection*, pages 162–182, London, UK, 2000. Springer-Verlag.

[Luce & Raiffa 1957] Robert Duncan Luce et Howard Raiffa. *Games and Decisions*. J. Wiley, New York, 1957.

[Mé 1994] Ludovic Mé. « *Audit de sécurité par algorithme génétique* ». Phd thesis, Université de Rennes 1, Rennes, France, 1994.

[Manganaris *et al.* 2000] Stefanos Manganaris, Marvin Christensen, Dan Zerkle, et Keith Hermiz. « A data mining analysis of RTID alarms ». *Computer Networks : The International Journal of Computer and Telecommunications Networking*, 34(4) :571–577, 2000.

[Mann & Christey 1999] David E. Mann et Steven M. Christey. « Towards a Common Enumeration of Vulnerabilities ». Dans *Proceeding of the 2nd Workshop on Research with Security Vulnerability Databases*, January 1999.

[McHugh *et al.* 2000] John McHugh, Alan Christie, et Julia Allen. « Defending Yourself : The Role of Intrusion Detection Systems ». *IEEE Software*, 17(5) :42–51, September 2000.

[Mé 1998] Ludovic Mé. « GASSATA, a genetic algorithm as an alternative tool for security audit trail analysis ». Dans *First International Workshop on the Recent Advances in Intrusion Detection*, Belgium, September 1998.

[Michel & Mé 2001] Cédric Michel et Ludovic Mé. « ADeLe : an Attack Description Language for Knowledge-based Intrusion Detection ». Dans *PIFIP/SEC*, pages 353–365, 2001.

[Milnor 1954] John Milnor. « *Games against nature* ». 1954.

[Monjardet 1978] Bernard Monjardet. « Axiomatiques et propriétés des quasi ordres ». *Mathématiques et Sciences Humaines*, 63 :51–82, 1978.

[Montanari 1974] Ugo Montanari. « Networks of constraints : Fundamental properties and applications to picture processing ». *Inf. Sci.*, 7 :95–132, 1974.

[Morin & Debar 2003] Benjamin Morin et Hervé Debar. « Correlation of Intrusion Symptoms : an Application of Chronicles ». Dans *Proceedings of the 6th International Conference on Recent Advances in Intrusion Detection (RAID'03)*, pages 94–112, 2003.

[Morin 2004] Benjamin Morin. « *Corrélation d'alertes issues d'outils de détection d'intrusions avec prise en compte d'informations sur le système surveillé* ». PhD thesis, INSA de Rennes, Février 2004.

[Morin *et al.* 2002] Benjamin Morin, Ludovic Mé, Hervé Debar, et Mireille Ducassé. « M2D2 : A formal Data Model for IDS Alert Correlation ». Dans *Recent Advances in Intrusion Detection (RAID'02)*, pages 115–137, October 2002.

[Mousseau 2003] Vincent Mousseau. « *Elicitation des préférences pour l'aide multicritère à la décision* ». Mémoire présenté en vue de l'obtention de l'habilitation à diriger des recherches, 2003.

[Naïm *et al.* 2007] Patrick Naïm, Pierre-Henri Wuillemin, Philippe Leray, Olivier Pourret, et Anna Becker. *Réseaux bayésiens*. 3 édition, 2007.

[Ning *et al.* 2002] Peng Ning, Yun Cui, et Douglas S. Reeves. « Constructing attack scenarios through correlation of intrusion alerts ». Dans *CCS '02 : Proceedings of the 9th ACM conference on Computer and communications security*, pages 245–254, New York, NY, USA, 2002. ACM.

[Ning *et al.* 2004] Peng Ning, Dingbang Xu, Christopher G. Healey, et Robert St. Amant. « Building Attack Scenarios through Integration of Complementary Alert Correlation Methods ». Dans *Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS'04*, pages 97–111, 2004.

[Orebaugh & Ramirez 2004] Angela D. Orebaugh et Gilbert Ramirez. *Ethereal Packet Sniffing*. Syngress Publishing, 2004.

[Paxson 1999] Vern Paxson. « Bro : a system for detecting network intruders in real-time ». *Computer Networks*, 31(23–24) :2435–2463, 1999.

[Pearl 1988] Judea Pearl. *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[Pinkas 1991] Gadi Pinkas. « Propositional Non-Monotonic Reasoning and Inconsistency in Symrnetric Neural Networks ». Dans *IJCAI'91*, pages 525–531, 1991.

[Porras & Neumann 1997] Phillip A. Porras et Peter G. Neumann. « EMERALD : Event Monitoring Enabling Responses to Anomalous Live Disturbances ». Dans *Prociding 20th NIST-NCSC National Information Systems Security Conference*, pages 353–365, 1997.

[Porras 1993] Phillip A. Porras. « STAT – A State Transition Analysis Tool For Intrusion Detection ». Rapport Technique, Santa Barbara, CA, USA, 1993.

[Prakken & Sartor 1996] Henry Prakken et Giovanni Sartor. « A dialectical model of assessing conflicting arguments in legal reasoning ». *Artificial Intelligence and Law*, 4 :331–368, 1996.

[Proctor 2000] Paul E. Proctor. *Practical Intrusion Detection Handbook*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000.

[Puterman 2005] Martin L. Puterman. *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 2005.

[Qi & Poole 1995] Runping Qi et David Poole. « New method for influence diagram evaluation ». *Computational Intelligence*, 11 :498–528, 1995.

[Qin 2005] Xinzhou Qin. « *A probabilistic-based framework for infosec alert correlation* ». PhD thesis, Atlanta, GA, USA, 2005.

[Quiggin 1982] John Quiggin. « A theory of anticipated utility ». *Journal of Economic Behavior & Organization*, 3(4) :323–343, December 1982.

[Quinlan] J Ross Quinlan. « Induction of Decision Trees ». *Machine Learning*, 1(1) :81–106.

[Quinlan 1993] J Ross Quinlan. *C4.5 : Programs for Machine Learning*. Morgan Kaufmann, 1993.

[Reiter 1980] Raymond Reiter. « A logic for default reasoning ». *Artificial Intelligence*, 13 :81–132, 1980.

[Rossi *et al.* 2004] Francesca Rossi, Kristen Brent Venable, et Toby Walsh. « mCP Nets : Representing and Reasoning with Preferences of Multiple Agents ». Dans *AAAI'04*, pages 729–734, 2004.

[Roy 1985] Bernard Roy. *Méthodologie multicritère d'aide à la décision*. Economica, Paris, 1985.

[Sabhnani & Serpen 2004] Maheshkumar Sabhnani et Gursel Serpen. « Why machine learning algorithms fail in misuse detection on KDD intrusion detection data set ». *Intelligent Data Analysis*, 8(4) :403–415, 2004.

[Sahami *et al.* 1998] Mehran Sahami, Susan Dumais, David Heckerman, et Eric Horvitz. « A Bayesian Approach to Filtering Junk E-mail ». Dans *AAAI Workshop on Learning for Text Categorization*, July 1998.

[Savage 1951] Leonard Jimmie Savage. « The theory of statistical decision ». Dans *Journal of the American Statistical association*, pages 55–67, 1951.

[Savage 1954] Leonard Jimmie Savage. *The Foundations of Statistics*. John Wiley & Sons, New York, 1954.

[Sen 1986] Amartya Sen. Social choice theory. Dans K.J. Arrow et M.D. Intriligator, éditeurs, *Handbook of mathematical economics*, volume 3, pages 1073–1181. North-Holland, Amsterdam, 1986.

[Shachter & Peot 1990] Ross D. Shachter et Mark A. Peot. « Simulation Approaches to General Probabilistic Inference on Belief Networks ». Dans *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 221–234, Amsterdam, The Netherlands, 1990.

[Shachter 1986] Ross D. Shachter. « Evaluating influence diagrams ». *Operational Research Journal*, 34(6) :871–882, 1986.

[Shipley 1999] Greg Shipley. « Intrusion Detection, take two ». *Network Computing Journal*, 10(23) :44–48, 1999.

[Shoham 1987] Yoav Shoham. « Nonmonotonic Logic : Meaning and Utility ». Dans *Proceedings of the 10th International Joint Conference in Artificial Intelligence, IJCAI87*, pages 388–393. Morgan Kaufmann, San Francisco, 1987.

[Spirtes *et al.* 2001] Peter Spirtes, Clark Glymour, et Richard Scheines. *Causation, Prediction, and Search, Second Edition (Adaptive Computation and Machine Learning)*. The MIT Press, January 2001.

[Staniford *et al.* 2002] Stuart Staniford, James A. Hoagland, et Joseph M. McAlerney. « Practical automated detection of stealthy portscans ». *Journal of Computer Security*, 10(1-2) :105–136, 2002.

[SunSoft 1995] SunSoft. « SunSHIELD Basic Security Module Guide ». Mountain View,CA, 1995.

[Tan & Pearl 1994] Sek-Wah Tan et Judea Pearl. « Specification and Evaluation of Preferences Under Uncertainty ». Dans *KR*, pages 530–539, 1994.

[Templeton & Levitt 2000] Steven J. Templeton et Karl Levitt. « A requires/provides model for computer attacks ». Dans *NSPW '00 : Proceedings of the 2000 workshop on New security paradigms*, pages 31–38, New York, NY, USA, 2000. ACM.

[Valdes & Skinner 2000] Alfonso Valdes et Keith Skinner. « Adaptive, model-based monitoring for cyber attack detection ». Dans *Recent Advances in Intrusion Detection (RAID 2000)*, pages 80–92. Springer-Verlag, 2000.

[Valdes & Skinner 2001] Alfonso Valdes et Keith Skinner. « Probabilistic Alert Correlation ». Dans *Recent Advances in Intrusion Detection (RAID 2001)*, numéro 2212 dans Lecture Notes in Computer Science, pages 54–68. Springer-Verlag, 2001.

[Vigna & Kemmerer 1999] Giovanni Vigna et Richard A. Kemmerer. « Netstat : A network-based intrusion detection system ». *Journal of Computer Security*, 7 :37–71, 1999.

[Vigna 2003] Giovanni Vigna. « A Topological Characterization of TCP/IP Security ». Dans *Proceedings of the 12 th International Symposium of Formal Methods Europe (FME'03)*, pages 914–939, septembre 2003.

[Vigna *et al.* 2000] Giovanni Vigna, Steven Eckmann, et Richard Kemmerer. « Attack Languages ». Dans *Proceedings of the IEEE Information Survivability Workshop*, 2000.

[von Neumann & Morgenstern 1947] John von Neumann et Oskar Morgenstern. *Theory of games and economic behaviour*. Princeton University Press, 2nd edition, Princeton, 1947.

[von Wright 1963] Georg Henrik von Wright. *The logic of preferences*. Edinburgh University Press, Edinburgh, 1963.

[Wald 1950] Abraham Wald. *Statistical Decision Functions*. New York, Wiley and Sons, 1950.

[Walters 1995] Rex Walters. « Tracking Hardware Configurations in a Heterogeneous Network with syslogd ». Dans *LISA '95 : Proceedings of the 9th USENIX conference on System administration*, pages 241–246, Berkeley, CA, USA, 1995. USENIX Association.

[Wellman & Doyle 1992] Michael P. Wellman et Jon Doyle. « Modular utility representation for decision-theoretic planning ». Dans *Proceedings of the first international conference on Artificial intelligence planning systems*, pages 236–242, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

[Wicker & Doyle 2007] Andrew W. Wicker et Jon Doyle. « Interest-Matching Comparisons using CP-nets ». Dans *AAAI,07*, pages 1914–1915, 2007.

[Wilkinson 2007] Darren J. Wilkinson. « Bayesian methods in bioinformatics and computational systems biology ». *Journal of Briefings in bioinformatics*, 8(2) :109–116, April 2007.

[Wilson 2004] Nic Wilson. « Extending CP-nets with stronger conditional preference statements ». Dans *Proceedings of (AAAI'04)*, pages 735–741, 2004.

[Xuren & Famei 2006] Wang Xuren et He Famei. « Improving Intrusion Detection Performance Using Rough Set Theory and Association Rule Mining ». Dans *ICHIT '06 : Proceedings of the 2006 International Conference on Hybrid Information Technology*, pages 114–119, Washington, DC, USA, 2006. IEEE Computer Society.

[Yu & Frincke 2007] Dong Yu et Deborah Frincke. « Improving the quality of alerts and predicting intruder's next goal with Hidden Colored Petri-Net ». *Journal of Computer Networks*, 51(3) :632–654, 2007.

[Zadeh 1999] Lotfi Askar Zadeh. « Fuzzy sets as a basis for a theory of possibility ». *Fuzzy Sets and Systems*, 100(supp.) :9–34, 1999.

[Zhou *et al.* 2005] Lina Zhou, Jinjuan Feng, Andrew Sears, et Yongmei Shi. « Applying the Naïve Bayes Classifier to Assist Users in Detecting Speech Recognition Errors ». Dans *Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05)*, Washington, DC, USA, janury 2005.