Techniques d'optimisation pour la fouille de données

Dominique Francisci

UNIVERSITE DE NICE-SOPHIA ANTIPOLIS - UFR SCIENCES

Ecole Doctorale STIC
(Sciences et Technologies de l'Information et de la Communication)

Département Informatique

THESE

Présentée pour obtenir le titre de

Docteur en Sciences
de l'Université de Nice-Sophia Antipolis

Spécialité : Informatique

par

Dominique FRANCISCI

Techniques d'optimisation pour la fouille de données

Soutenue publiquement le jeudi 11 mars 2004
devant le jury composé de :

M^r Jean-Louis CAVARERO	*Professeur à l'UNSA*	Président
M^r Marco TOMASSINI	*Professeur à l'Université de Lausanne*	Rapporteur
M^r Djamel A. ZIGHED	*Professeur à l'Université Lyon II*	Rapporteur
M^me Martine COLLARD	*Maître de Conférence à l'UNSA*	Directrice
M^r Manuel CLERGUE	*Maître de Conférence à l'UNSA*	Examinateur
M^r Jean-Fabrice LEBRATY	*Professeur à l'UNSA*	Examinateur
M^r Albert FOUCHER	*Président du CPP SFR-Cegetel*	Invité

Remerciements

Cette thèse a été financée par la Fondation d'Entreprise SFR-Cegetel et a été réalisée au sein du projet EXeCO [1] (EXtraction de connaissances à partir des données et analyse et COnception des systêmes d'information) au Laboratoire Informatique Signaux et Systèmes de Sophia Antipolis [2], unité mixte de recherche du Centre National de la Recherche Scientifique [3] et de l'Université de Nice-Sophia Antipolis [4].

Je tiens à remercier les personnes suivantes :

- Madame Martine COLLARD, ma directrice de thèse, qui m'a permis d'effectuer cette thèse sous sa direction.

- Monsieur Jean-Louis CAVARERO, sans qui cette thèse n'aurait pu être entreprise.

- Messieurs Marco TOMASSINI et Djamel A. ZIGHED, qui m'ont fait l'honneur de lire mes travaux et d'accepter de les rapporter.

- Messieurs Albert FOUCHER, Manuel CLERGUE et Jean-Fabrice LEBRATY, qui m'ont fait le plaisir d'être les autres membres du jury.

[1] Site WEB du projet EXeCO : http ://www.i3s.unice.fr/~mcollard/execo
[2] Site WEB du laboratoire I3S : http ://www.i3s.unice.fr
[3] Site WEB du CNRS : http ://www.cnrs.fr
[4] Site WEB de l'UNSA : http ://www.unice.fr

A mes parents.

Résumé

Les développements accrus des technologies numériques ont engendré depuis quelques années, des volumes de données extrêmement importants, qui peuvent receler des informations utiles pour les organismes qui les ont produites. Ce constat a donné naissance à un champ d'exploration : l'extraction de connaissances à partir des données, également appelée fouille de données, ou encore data mining dans la terminologie anglo-saxone. L'extraction de connaissances à partir des données désigne le processus non-trivial d'extraction d'informations implicites, précédemment inconnues et potentiellement utiles enfouies dans les données.

La fouille de données comprend globalement cinq phases qui sont la sélection des données, leur pré-traitement, leur transformation, l'extraction de modèles ou motifs et enfin l'évaluation de connaissances extraites. L'ensemble des travaux que nous présentons dans ce mémoire, s'inscrit dans ce contexte et en particulier dans la phase d'extraction de modèles proprement dite. Nous nous intéressons d'une part aux connaissances exprimées sous la forme de règles de dépendance et d'autre part à la qualité de ces règles. Une règle de dépendance représente une implication conditionnelle entre ensembles d'attributs du jeu de données et la qualité d'une règle est relative à sa précision, sa compréhensibilité, son utilité, et son intérêt pour l'utilisateur final. Les travaux existants relatifs aux mesures de qualité des règles de dépendance se basent généralement sur une étude analytique. Les algorithmes standards mis en œuvre ont pour but de rechercher les meilleurs modèles ; par exemple un algorithme d'induction d'arbres de décision génère le meilleur classifieur sur les instances traitées et ce en choisissant à chaque niveau de l'arbre, l'attribut le plus discriminant du jeu de données. En recherche de règles d'association, les algorithmes réalisent un parcours de l'espace de recherche afin de ne retenir que les modèles fournissant la meilleure description du jeu de données. Derrière ces processus se cache en fait une véritable problématique d'optimisation qui n'est pas explicitement exprimée.

La fouille de données considérée selon ce point de vue constitue l'essentiel de notre approche. Nous considérons la recherche des règles de dépendance les plus intéressantes comme étant un problème d'optimisation dans lequel la qualité d'une règle est quantifiée par le biais d'une ou de plusieurs mesures de modèles. Notre approche est expérimentale ; il ressort en effet que la plupart des mesures observées présentent des propriétés différentes suivant le jeu de données et de fait, une approche analytique est difficilement envisageable sans fixer certains paramètres. Ainsi, étant donné que nous considérons la fouille de données sous l'angle de l'optimisation, il est nécessaire d'étudier les espaces de recherche induits par les mesures ainsi que les algorithmes de recherche dans ces espaces. Nous observons les variations relatives de mesures évaluées simultanément ; certaines d'entre elles sont antagonistes ce qui ne permet pas d'obtenir

« la » meilleure règle ; il faut alors considérer un ensemble de compromis satisfaisants. Nous présentons dans ce mémoire, plusieurs approches permettant de mettre en évidence des particularités des espaces de recherche et nous apportons des solutions par le biais de l'approche évolutionnaire. Notre objectif est donc d'extraire des modèles à base de règles de dépendance en mettant en œuvre des algorithmes d'extraction permettant l'optimisation d'une mesure ou d'un ensemble de mesures de qualité. Nous nous sommes orientés vers une approche évolutionnaire et en particulier vers les algorithmes génétiques, métaheuristique éprouvée et très efficace sur de vastes espaces de recherche comme ils se présentent en fouille de données. D'autre part, ces algorithmes permettent de mettre en œuvre l'optimisation multi-objectifs et donc de prendre en compte plusieurs mesures de qualité et d'extraire les modèles représentant les meilleurs compromis vis à vis des critères choisis par l'utilisateur.

Mots clés : extraction de connaissances à partir des données, base de données, algorithme évolutionnaire, règle de dépendance, mesure de qualité, optimisation multi-critères.

Abstract

The increased development of numerical technologies have generated for a few years, huge volumes of data, which can conceal useful information for the organizations which produced them. This situation gave rise to a field of exploration : the knowledge discovery in data bases also called data mining. The data mining process indicates the non obvious process of extracting implicit informations, previously unknown and potentially useful hidden into the data.

A standard knowledge discovery process includes five steps : data selection, data preprocessing, data transformation, data mining and finally evaluation of the extracted knowledge. The work which we present in this thesis, focuses on data mining step. We are interested in a kind of information expressed as a dependency rule. A dependency rule is a conditional implication between sets of attributes over the data set and the interestingness of a rule is related to its precision, its comprehensibility, its utility, its aptitude to surprise the end-user. We compare the interest to the quality of the rules. Existing works related to measures of dependency rules quality tend for the majority to provide formalization of measures. The purpose of standard data mining algorithms is generally to find the best model or pattern. For example, a decision tree induction algorithm generates the best classifier on the data by choosing for each level of the tree, the most discriminating attribute. In the process of association rule extraction, a standard algorithm tends to carry out an exhaustive course of the space of research in order to retain only the rules providing the best description of the data set. In fact, behind these two processes, there is an optimization problem which is not explicitly expressed.

The data mining process considered according to this point of view constitutes the essence of our approach. This one is experimental and we consider interestingness of dependency rules as being a problem of optimization. The interestingness of a rule is quantified by the mean of a measure or a set of measures. Indeed, it arises that these measures have a different behavior according to the data set involved, and so, an analytical approach is not easily possible. Thus, since we consider the data mining process under the angle of optimization, it is necessary to study the search space induced by measures as well as search algorithms associated with the analysis of these spaces. Moreover, it also arises that some of these measures, when they are considered simultaneously, present antagonisms ; thus obtaining "the" best rule is not possible and it is necessary to consider a set of good tradeoffs. We present in this thesis, several approaches allowing to highlight characteristics related to search spaces and we bring solutions by the mean of the evolutionary approach. Our objective is thus to extract rule based models containing dependency rules by the mean of data mining algorithms able to optimize a single measure or a set of quality measures. We directed ourselves towards an evolutionary approach and in particular towards the genetic algorithms which

are a well-known metaheuristic for its effective in presence of huge search spaces. In addition, they make possible to implement multi-objective optimization and thus to take into account several measures of quality in order to extract the best models relatively to the measures chosen by the user.

Keywords : knowledge discovery in data bases, data base, evolutionary algorithm, dependency rule, quality measure, multi-criteria optimization.

Table des matières

Chapitre 1

Introduction

Sommaire

Les avancées relatives aux technologies numériques ont donné naissance, depuis quelques années, à des volumes de données extrêmement importants, qui renferment potentiellement des informations utiles pour les divers organismes à l'origine de ces données. Cet état de fait a donné lieu à un nouveau champ d'exploration : celui de *l'extraction automatique de connaissances à partir des données* ou encore *fouille de données*. La fouille de données également appelée « data mining » dans la terminologie anglo-saxone a pour objectif de produire des « modèles » compacts et compréhensibles pour l'utilisateur et ce de façon automatique à partir de grands volumes de données. Il est d'usage de dire que la fouille de données est une extension des statistiques par des moyens plus automatisés et plus puissants. En effet, statistiques et fouille de données ont le même objectif, qui est de réaliser des « modèles » compacts et compréhensibles rendant compte des relations liant la description d'une situation à un résultat (ou un jugement) concernant cette description. L'hypothèse implicite est bien sûr que le résultat, la mesure ou le jugement que nous essayons de modéliser dépend effectivement des éléments de description dont nous disposons.

Une différence essentielle entre les deux domaines est que les techniques de fouille de données tentent de construire le modèle de manière automatique alors que les techniques statistiques « classiques » requièrent d'être maniées et guidées par un statisticien professionnel, celui-ci ayant déjà une idée - peut-être préconçue - des « hypothèses de dépendance à formuler ».

Une autre différence est relative aux volumes de données traités. En effet, ceux-ci ne cessent d'augmenter de par l'accroissement de la taille des unités de stockage et de par l'activité des organismes à l'origine de ces données. Une des conséquences de ce stockage massif est qu'il est éventuellement possible d'en extraire des connaissances cachées, utiles pour l'aide à la décision, la gestion, le contrôle de processus, etc. Les connaissances extraites sont des règles, des concepts, des régularités, des anomalies et des modèles qui sont utiles, intéressants et compréhensibles du point de vue de l'utilisateur [Fayy96b]. Dans cette thèse, nous nous intéressons à l'extraction de modèles à base de règles de dépendance.

Dans un environnement de plus en plus concurrentiel, les entreprises ont besoin de transformer le plus rapidement possible ces masses de données en produit fini : la connaissance. Ainsi, le défi de la fouille de données est souvent celui de la productivité face à cette croissance du volume de données. Les techniques de fouille de données apportent un gain important tant en performance qu'en maniabilité ou en temps de travail. La possibilité de réaliser ses propres modèles statistiques par « soi-même » sans besoin de sous-traiter ou de consulter un statisticien apporte une grande liberté aux utilisateurs opérationnels.

D'une manière générale, les techniques de fouille de données ont une raison d'être partout où il existe de nombreuses informations et où les processus peuvent être améliorés, c'est-à-dire dans de nombreux secteurs d'activité. L'extraction des informations à partir des bases de données présente un intérêt majeur pour le secteur industriel. En effet, les informations recueillies peuvent, par exemple, faire ressortir un comportement chez des clients ce qui peut permettre d'améliorer les techniques de vente. Dans la grande distribution, elles peuvent mettre en évidence des corrélations entre produits en analysant l'importante quantité d'information disponible, ce qui peut préciser de nouveaux axes de recherche.

Les travaux relatifs à ce domaine sont motivés d'une part par l'évolution rapide des techniques d'acquisition des données et d'autre part par l'accroissement constant des supports de stockage. Si les nouvelles bases de données permettent de stocker des volumes d'informations toujours plus importants, à des coûts de plus en plus faibles, force est de constater que les technologies d'analyse et de visualisation n'ont pas évolué en même temps.

Ce chapitre est organisé comme suit : nous présentons dans la section suivante les tâches les plus courantes en fouille de données. Ensuite, nous introduisons la problématique ainsi que les motivations de nos travaux. Nous concluons ce chapitre par la présentation de notre contribution et de l'organisation du mémoire.

1.1 Algorithmes standards en fouille de données

Cette section introduit les trois tâches standards de la fouille de données. Il s'agit de la classification, de la modélisation prédictive et de la recherche de règles d'association. Dans le cadre de cette thèse nous nous intéressons aux règles extraites en modélisation prédictive aussi bien qu'en recherche d'associations.

Classification

La classification est également appelée apprentissage non-supervisé. En classification, le but est de partitionner l'ensemble des données en un certain nombre de classes homogènes. Cela signifie que les classes doivent contenir des données qui partagent un haut degré de similarité ; le but est de maximiser la similarité à l'intérieur d'une classe et de minimiser la similarité entre classes. La similarité des objets est mesurée en termes de distance entre les données. La difficulté réside dans la recherche de modèles qui optimisent non seulement les distances intra-classe et inter-classes, mais également le nombre de classes qui doit rester relativement faible. Contrairement à la modélisation prédictive que nous présentons ci-dessous, où les classes sont connues dès le départ, dans la tâche de classification, les classes ne sont pas connues par le système au début du processus.

Modélisation prédictive

La modélisation prédictive permet de classer des instances d'un jeu de données dans un ensemble de classes connues. Cette modélisation est également explicative car elle permet « d'expliquer » la valeur d'un attribut par rapport aux autres. Les algorithmes construisent un modèle de classement à partir d'un ensemble d'instances appelé « ensemble d'apprentissage ». Dans cette tâche, les classes sont connues dès le début du processus.
Les travaux sur la modélisation prédictive, comme pour la classification non supervisée, ne sont pas spécifiques du domaine de la fouille de données ; dans le cadre de l'apprentissage automatique, on trouve de nombreuses propositions d'algorithmes de classification basés sur un réseau de neurones ou une méthode d'induction d'arbres de décision ou encore un classifieur Bayésien. Nous citons ici les méthodes d'induction d'arbres de décision comme référence, car ces structures se traduisent simplement sous forme de règles et intègrent aussi bien des variables prédictives discrètes que continues. La construction de l'arbre de décision consiste à utiliser les variables prédictives pour partitionner récursivement l'ensemble des données en sous-ensembles de manière à maximiser la pureté de la partition.
Ce type de méthode génère le meilleur classifieur sur les instances traitées par le biais de l'heuristique suivante : à chaque niveau de l'arbre de décision en cours de construction, l'algorithme d'induction choisit l'attribut du jeu de données le plus discriminant. La racine de cet arbre « contient » l'ensemble des instances du jeu. Ses fils contiennent

l'ensemble de ces instances partitionnées selon la valeur de l'un des attributs. Chacun de ces fils divise à son tour l'ensemble des instances qu'il contient dans l'ensemble de ses fils et ainsi de suite, à chaque fois selon l'un des attributs. L'ensemble des feuilles de l'arbre de décision représente donc un partitionnement de l'ensemble des instances de la racine. Une fois l'arbre généré, il permet de « classer » n'importe quelle instance du jeu de données originel dans une feuille de l'arbre.
Afin d'obtenir des arbres de décision de taille réduite, l'élagage est mis en œuvre. Il consiste à supprimer les sous-arbres ne vérifiant pas une certaine condition qui repose sur le taux d'erreur. Cet élagage est appliqué après que l'arbre ait été développé au maximum.

Recherche d'associations

La recherche de règles d'association est une approche descriptive qui a pour but de construire un modèle à base de règles liant les données de façon significative entre elles. Une règle d'association est définie syntaxiquement comme une règle logique d'ordre 0 de la forme *si* A *alors* B où les propositions A et B sont des conjonctions d'expressions simples de comparaison sur les attributs du jeu de données.
Contrairement aux règles de classification, les règles d'association peuvent mettre en évidence des corrélations inattendues entre les données du jeu de données. Cependant, le nombre de règles générées est souvent trop important pour être pertinent.

1.2 Problématique

Comme nous venons de le présenter dans la section précédente, chacune des tâches tend à déterminer « le meilleur » modèle sur le jeu de données traité. Ainsi, se cache derrière ces processus, un véritable problème d'optimisation combinatoire non explicitement exprimé. En effet, comme on peut le voir dans le schéma de la figure 1.1, la fouille de données peut être considérée comme un problème d'optimisation. En effet, il s'agit de déterminer un ou plusieurs modèles parmi un ensemble de modèles. Il est donc nécessaire d'en évaluer la qualité ce qui est réalisé au moyen de critères, lesquels doivent être optimisés.

Dans ce mémoire, nous nous intéressons uniquement aux modèles à base de règles de dépendance et ce d'un point de vue optimisation et indépendamment des différentes techniques que nous venons de présenter. Une règle de dépendance de la forme $r : A \rightarrow C$ est un lien de causalité entre deux sous-ensembles A et C d'attributs d'un jeu de données.

Nous considérons donc la fouille de données comme un problème d'optimisation. De fait, notre objectif est d'étudier d'une part les espaces de recherche associés aux mesures considérées et d'autre part les algorithmes ad hoc permettant de parcourir ces

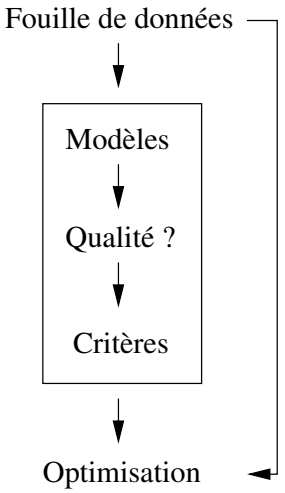


FIG. 1.1 – *La fouille de données considérée sous l'angle de l'optimisation.*

espaces. Enfin, notre optique étant d'optimiser un ensemble de mesures de qualité que les règles doivent satisfaire, nous considérons une catégorie d'algorithmes d'optimisation, celle des algorithmes évolutionnaires et en particulier les algorithmes génétiques. Ces derniers sont dotés de caractéristiques qui sont particulièrement bien adaptées au contexte dans lequel nous nous plaçons. En effet, ces algorithmes permettent :

- Un parcours plus efficace des espaces de recherche en présence, grâce à l'évolution d'une population de solutions potentielles et non d'une seule solution comme il est généralement d'usage dans les algorithmes standards. En effet, les jeux de données en fouille de données et ceux à partir desquels nous recherchons des règles sont associés à de tels espaces de recherche de très grandes cardinalités. Considérons à titre d'exemple la recherche de règles de prédiction sur un jeu de données constitué de M attributs catégoriels. Le $i^{ème}$ attribut est associé a un domaine D_i constitué de $|D_i|$ valeurs auquel une valeur particulière est ajoutée afin de spécifier si l'attribut est utilisé ou pas dans la règle de prédiction. Soit k tel quel $1 < k < M$ l'indice de l'attribut à prédire. Le nombre R de règles associées à ce jeu de données pour la prédiction d'une valeur de l'attribut d'indice k (la classe) est :

$$R = \left(\prod_{i=1}^{k-1} |D_i| \right) \times \left(\prod_{j=k+1}^{M} |D_j| \right)$$

- L'évaluation globale des règles. Contrairement aux algorithmes standards qui ne possèdent pas cette caractéristique, celle-ci se révèle être cruciale dans la recherche de règles comportant des corrélations entre certains attributs.

- L'expression de façon aisée des solutions (des règles) par le biais d'un formalisme de haut niveau ce qui constitue un des objectifs d'une tâche de fouille de données.

- La mise en œuvre de mécanismes d'optimisation multi-objectifs, c'est-à-dire la considération et l'optimisation simultanée de plusieurs mesures de qualité.

Les algorithmes standards tendent, de par leur architecture, à écarter de vastes sous-ensembles des espaces de recherche en présence et, de fait, risquent d'éliminer des règles « intéressantes » selon d'autres critères objectifs (d'intérêt par exemple) ou selon des critères subjectifs exprimés par l'utilisateur. En effet, si l'on considère les algorithmes d'induction d'arbres de décision dérivés de l'algorithme C4.5 [Quin96], il s'agit, comme nous l'avons évoqué précédemment, de partitionner l'espace de recherche selon les attributs les plus discriminants. Ainsi, dans l'hypothèse où des règles couvrant peu d'exemples sont recherchées, ce type d'algorithme n'est pas en mesure d'apporter une solution adéquate étant donné que cette catégorie de règles est souvent éliminée dans la phase d'élagage. Pour leur part, les algorithmes fondés sur la base de l'algorithme APRIORI [Agra94] procèdent en deux étapes dont la première a pour objectif de filtrer les règles selon leur valeur de *Support*. De fait, ceci écarte également un certain nombre de règles qui pouvaient se révéler « intéressantes » selon un autre critère.
Fournir d'une part un modèle des plus précis et d'autre part intéressant, ne sont pas les seules qualités attendues d'un algorithme d'extraction de connaissances. L'utilisateur doit pouvoir facilement interpréter les résultats. Ainsi, on recherche plutôt des algorithmes fournissant des modèles lisibles. Cependant, en fouille de données, on constate le fait suivant : plus un modèle est lisible, moins l'algorithme qui le fournit est robuste. En d'autres termes, un algorithme robuste a tendance à générer un modèle peu lisible. C'est le cas par exemple des réseaux de neurones qui sont reconnus comme étant une des catégories d'algorithmes les plus robustes, mais les modèles engendrés pas ce biais sont très peu lisibles.
Un autre problème concerne l'interaction entre attributs. Cet aspect est crucial lorsque les données sont fortement corrélées, car les algorithmes standards génèrent alors des modèles constitués d'un trop grand nombre de règles pour l'utilisateur.
Enfin, peu d'algorithmes en fouille de données sont à même de pouvoir considérer simultanément plusieurs mesures de qualité des règles, ce point est néanmoins essentiel. Afin d'apporter des solutions à ces aspects, nous nous sommes orientés vers les algorithmes évolutionnaires. Nous présentons dans la section suivante, les approches que nous avons choisies dans le contexte de la fouille de données et selon la problématique que nous venons de présenter.

1.3 Motivations et contributions

Face aux nombreuses mesures de qualité proposées et à la nécessité d'obtenir des modèles les plus précis et surtout les plus intéressants, il convient d'étudier et de com-

parer ces mesures entre elles. Un certain nombre de résultats ont déjà été publiés ; ils sont en général basés sur des études analytiques des mesures.
Notre approche est expérimentale et repose sur le constat suivant obtenu à l'issue d'analyses comparatives [Fran02, Fran03a] : d'une part les propriétés et variations des mesures de qualité dépendent fortement des caractéristiques des jeux de données et d'autre part certaines sont liées par des corrélations positives ou négatives qu'il est important de connaître avant de les combiner. Notre choix s'est orienté vers l'emploi d'algorithmes évolutionnaires pour plusieurs raisons. La première est que ceux-ci permettent un parcours très large des espaces de recherche. Grâce à leur parallélisme implicite, ils permettent de manipuler à chaque instant un ensemble de solutions contrairement aux méthodes standards. Enfin, ils permettent une expression aisée des mesures de qualité au sein de leur fonction d'adaptation.
Il s'agit dans cette thèse de mettre en lumière l'apport de l'emploi de ces techniques dans un contexte d'optimisation d'une ou d'un ensemble de mesures de qualité de règles de dépendance. Notre but est de nous focaliser sur l'optimisation de la qualité des règles extraites en particulier lorsque plusieurs critères de sélection sont fixés.
Dans cette optique, il est nécessaire d'étudier d'une part les caractéristiques des espaces de recherche des règles, et d'autre part l'impact de la variation des paramètres de l'algorithme évolutionnaire sur son comportement et la qualité des règles extraites.
Dans le contexte de la fouille de données, ce qui fait généralement la faiblesse des algorithmes évolutionnaires concerne l'application de la fonction d'adaptation. En effet, il est nécessaire pour ces algorithmes de parcourir l'ensemble des instances du jeu de données pour évaluer un individu chaque fois que celui-ci a subi une modification ou bien lorsqu'il est créé. Or, les instances souvent très nombreuses sont généralement stockées dans des mémoires de masse ce qui implique une durée de traitement relativement longue. Globalement, deux approches existent pour réduire ce coût : la première consiste à utiliser le parallélisme et l'autre à échantillonner les instances du jeu de données. Nous proposons une méthode alternative reposant sur une structure de cache et permettant d'accroître l'application de la fonction d'adaptation dans ce contexte.

L'étude de l'espace de recherche nous a conduit à effectuer plusieurs analyses comparatives de mesures sur plusieurs jeux de données à l'aide de deux algorithmes d'approximation des espaces de recherche ; ces derniers sont étudiés selon plusieurs approches que nous présentons dans le mémoire. Il ressort de ces études que d'une part les comportements des mesures sur les jeux de données sont fortement influencés par les caractéristiques de ceux-ci et d'autre part, que ces mêmes mesures, lorsqu'elles sont considérées de façon simultanée, présentent des antagonismes. Le premier constat nous a conduit à adopter une approche expérimentale. Nous avons effectué de multiples observations d'algorithmes génétiques basées sur une mesure de qualité et ce pour de nombreux paramétrages des algorithmes afin de mettre en évidence leurs comportements. Ceci nous a conduit a envisager la prise en compte de plusieurs mesures et donc d'opter pour une approche multi-objectifs.

Nous avons développé un algorithme génétique (et générique) mono et multi-objectifs pouvant être appliqué à tout type de jeu de données. Nous avons étudié l'influence du paramétrage de l'algorithme sur la nature des résultats obtenus.

Cet algorithme génétique, nous a permis de déterminer les ensembles de solutions « optimales » [1] sur plusieurs jeux de données aux caractéristiques très différentes.

Dans la suite de ce mémoire, nous faisons référence à un algorithme d'induction d'arbres de décision et à un algorithme de recherche de règles d'association appartenant au système WEKA [2].

1.4 Organisation du mémoire

Le mémoire est organisé selon quatre chapitres dont nous présentons sommairement les contenus ci-dessous.

Le chapitre 2 présente les différentes étapes du processus d'extraction automatique de connaissances à partir des données. Les différentes tâches sont abordées et l'accent est mis sur les tâches d'apprentissage supervisé ainsi que sur celle de la recherche de règles d'association.

Le chapitre 3 concerne la présentation des mesures de qualité des règles que nous avons considérées dans cette thèse ; l'approximation et l'étude des espaces de recherche y sont présentées. Deux points sont abordés : la prise en compte d'une mesure pour la recherche de règle de dépendance et la prise en compte de plusieurs mesures. Concernant ce dernier aspect, nous avons considéré le cas de deux mesures afin de faciliter l'appréciation des résultats.

Le chapitre 4 est relatif à la présentation de la métaheuristique que nous considérons : celle des algorithmes évolutionnaires et en particulier des algorithmes génétiques. Nous présentons un état de l'art des travaux existants mettant en œuvre des méthodes évolutionnaires en fouille de données. Nous présentons dans ce chapitre les résultats obtenus à l'aide d'une version de l'algorithme génétique que nous avons développée et prenant en compte une seule mesure de qualité pour extraire les règles de dépendance.

Le chapitre 5 est consacré à la prise en compte de plusieurs mesures de qualité pour l'extraction de règles de dépendance par le biais d'un algorithme évolutionnaire. Nous présentons dans ce chapitre le problème de l'optimisation multi-objectifs ainsi que quelques travaux existants relatifs à l'emploi d'algorithmes évolutionnaires multi-objectifs en fouille de données. Enfin, nous présentons l'essentiel de notre approche

[1] Nous verrons dans le chapitre 5 selon quel point de vue ces solutions sont dites « optimales ».

[2] WEKA est un ensemble d'outils permettant de couvrir toutes les étapes du processus de découvrerte de connaissances à partir des données.

ainsi que les résultats que nous avons obtenus avec cette technique.

Le chapitre 6 conclut le mémoire et fournit un certain nombre de perspectives dans la continuité de nos travaux.

Chapitre 2

Fouille de données : techniques standards et limites

Sommaire

Dans ce chapitre nous détaillons le processus d'extraction de connaissances à partir des données et les principales tâches relatives à cette discipline à savoir la classification, la recherche d'associations et l'apprentissage supervisé.

2.1 Le processus de la fouille de données

U. M. Fayyad et al. [Fayy96a] donnent la définition suivante : *« La fouille de données est un processus semi-automatique et itératif de plusieurs phases allant de la sélection et la préparation des données jusqu'à l'interprétation des résultats »*. Les différentes étapes du processus de découverte de connaissances sont souvent représentées par un schéma comme celui de la figure 2.1.

1. La *sélection des données* constitue la première étape du processus et permet de ne garder parmi les données disponibles provenant souvent de plusieurs bases, que celles dont l'utilisateur a besoin, afin d'en extraire les informations cachées.

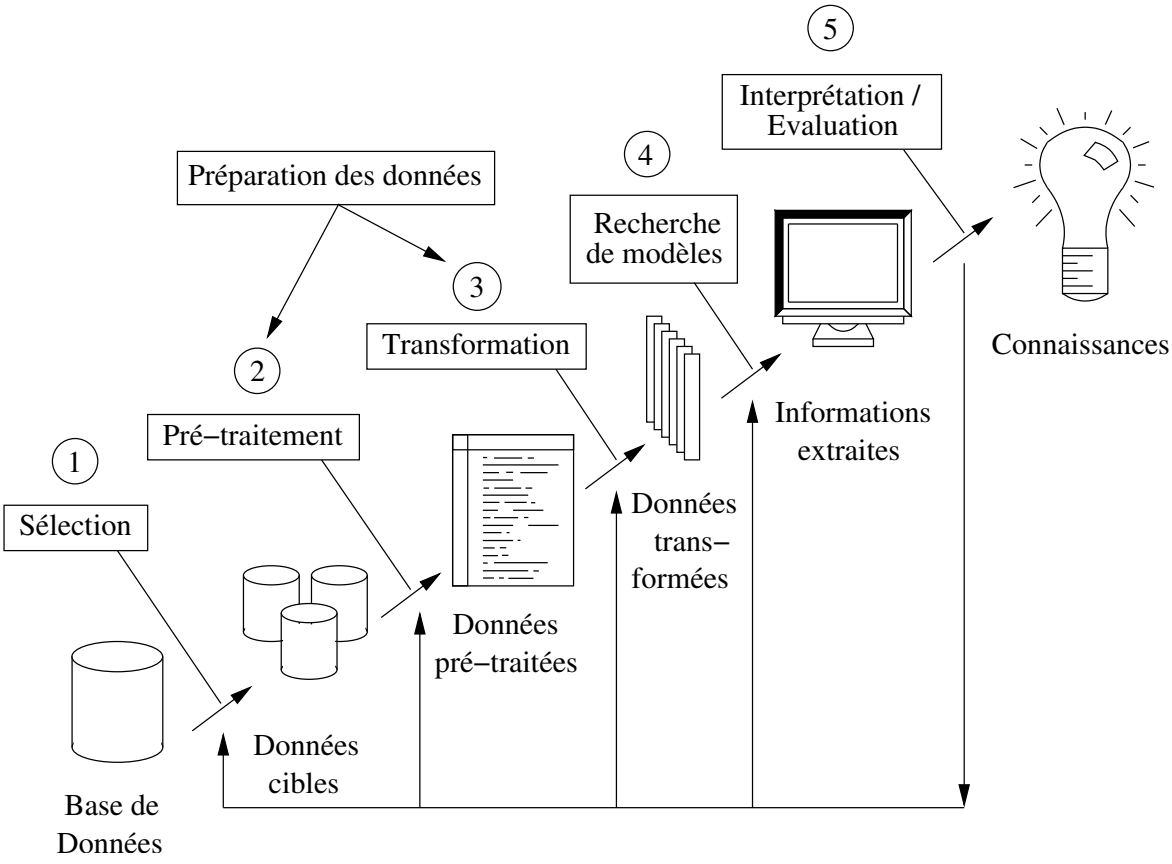


FIG. 2.1 – *Des données aux connaissances : les cinq étapes du processus d'extraction de connaissances à partir des données.*

2. Le *pré-traitement* constitue la seconde étape du processus et représente une phase de nettoyage des données qui a pour objectif de détecter les valeurs aberrantes, les valeurs manquantes et les valeurs nulles.

3. La *transformation des variables* constitue la troisième étape du processus et permet de présenter les données sous la forme attendue par l'algorithme d'extraction de connaissances.

4. L'*application d'algorithmes de recherche de modèles* représente la clé de voûte du processus et constitue la quatrième étape de celui-ci. Elle permet de mettre en évidence les informations sous-jacentes qui sont cachées dans les données.

5. L'*interprétation et l'évaluation des informations extraites* constitue la cinquième et dernière étape du processus et a pour objectif de produire de la connaissance sur le domaine d'étude en s'assurant de la validité de la sémantique des modèles

extraits. Il s'agit de supprimer les informations inutiles ou encore redondantes et de transformer les connaissances intéressantes en connaissances compréhensibles par l'utilisateur final. A ce stade, un retour aux autres étapes est possible. Néanmoins, force est de constater que le choix de l'étape à laquelle retourner n'est pas aisé. Par conséquent, cette cinquième étape est souvent la plus difficile en pratique.

Les étapes relatives au pré-traitement et à la transformation des variables représentent la phase de préparation des données.

Le meiller moyen de créer un modèle est de rechercher dans le passé des événements similaires. Il faut donc constituer, à partir de la mémoire de l'entreprise, cette base d'informations qui permet de construire le modèle. La récupération des données peut être plus ou moins facilitée par les technologies en place ; en particulier l'existence d'un entrepôt de données est importante.
La première phase de sélection constitue, lorsque les systèmes informatiques sources sont peu documentés et hétérogènes, une charge de travail considérable qui peut représenter jusqu'à 80% de la charge de travail globale du processus de fouille de données. De plus, certaines études nécessitent l'organisation d'un plan de collecte des données : entretiens qualitatifs, création de programmes pour « intercepter » des données qui ne font que « transiter » dans le système d'informations. Face au sentiment de perte de temps et d'inutilité éprouvé par les « clients » du projet fouille de données pendant cette étape, le responsable du projet doit mettre en œuvre une politique d'animation et de suivi de la collecte.
Le processus que nous venons de décrire est globalement le même quelque soit le type de modèle à découvrir, ou en d'autres termes, quelque soit la tâche considérée. La section suivante donne une présentation détaillée des tâches de prédiction et de description qui nous intéressent pour l'extraction de règles de dépendance.

2.2 Modèles et tâches

Nous présentons particulièrement dans cette section, les techniques mises en œuvre dans la phase d'extraction proprement dite, qui agit sur des données nettoyées et préparées. Dans cette phase, les algorithmes mis en œuvre visent à extraire différents types de modèles. Nous détaillons essentiellement dans cette section l'extraction des modèles à base de règles.
Les statistiques et plus particulièrement la fouille de données permettent d'explorer les relations entre données ainsi que leur importance. La force d'une relation peut être calculée par le coefficient de corrélation. La recherche des corrélations permet de distinguer dans quelle mesure différents éléments interagissent. La recherche de la cause des relations entre de nombreuses variables est assez complexe. En effet, si l'on considère seulement deux variables, il est assez facile de voir s'il y a ou non corrélation entre elles.

Par contre, l'intervention de variables tierces peut perturber cette recherche. Ainsi, la corrélation entre les variables A et B peut s'expliquer de diverses manières : il est possible que cette relation soit causale de A vers B ou de B vers A ou qu'une variable tierce C intervienne, que la modification de la variable C engendre à la fois la modification de A et la modification de B.
Les statistiques, sont limitées par rapport au nombre de variables à analyser dans de grandes bases de données où le nombre possible de relations entre variables est trop important pour les tester une à une. Il est donc essentiel d'utiliser aussi des techniques de recherche « intelligente », même si, dans certains cas, les techniques statistiques sont utilisées pour s'assurer de l'importance de la relation.
Dans la suite, nous considérons la recherche de règles de dépendance selon deux approches différentes :

- L'approche descriptive.
- L'approche prédictive.

2.2.1 Approche descriptive

Classification

La classification est caractérisée comme une activité d'apprentissage non supervisé. Le but est de fractionner l'ensemble hétérogène d'objets donnés en un certain nombre de sous-ensembles plus homogènes, appelés classes. Cela signifie que les classes doivent contenir des objets qui partagent un haut degré de similarité (maximisation de la similarité intra-classe) et que ces classes doivent être suffisamment larges (minimisation de la similarité inter-classes).
Le problème de classification a été étudié dans les domaines des statistiques [Chee96], de l'analyse de données [Carp93] et de la fouille de données [Hart75, Genn90, Agra98, Jain99].
La classification par classes est un très bon outil lorsque l'on est confronté à un ensemble de données contenant de nombreuses variables et une structure interne complexe. De même, cette technique peut être utilisée même si l'on n'aboutit qu'à une seule classe. Prenons l'exemple des produits fabriqués dans une usine : les méthodes de détection des classes peuvent être utilisées sur un échantillon ne contenant que de bons produits fabriqués pour déterminer le poids et la forme de la « bonne » classe. Lorsqu'un produit s'écarte de la classe pour une raison quelconque, il devient suspect. Une fois que l'on trouve une classe forte, on va l'analyser pour comprendre sa singularité.
Quelles sont les caractéristiques des enregistrements de cette classe qui ont permis leur regroupement ? La manière la plus facile d'approcher cette question est de prendre la moyenne de chaque variable classe et de la comparer avec la moyenne de la même variable dans la population parente pour ensuite ordonner ces variables qui apportent le

plus de différences entre la classe et le reste du jeu de données, nous pourrons expliquer la singularité de cette classe.
L'avantage que représente la classification réside dans le fait qu'elle est une technique d'apprentissage non supervisée et peut être appliquée sans aucune connaissance préalable du jeu de données et donc demande peu de préparation des données en entrée. De même, la classification s'applique à plusieurs types de données grâce aux différentes mesures de distance.
Par contre, sa faiblesse réside dans la difficulté à choisir les bonnes formules de distance : les performances des algorithmes de classification dépendent fortement de la formule de similarité choisie au début. Ainsi, dans certaines situations ou les données sont de plusieurs types, il est très difficile de fabriquer une formule pour la distance. De même, cette formule est sensible aux paramètres initiaux qui ont une forte influence sur les classes obtenues en fin de processus.

Association

Le problème de l'extraction de règles d'association dans le cas de le fouille de données a été introduit par R. Agrawal et al. [Agra93b]. La recherche de règles d'association a été developpée à l'origine pour l'analyse de bases de données de transactions de ventes, et a pour but de découvrir des relations significatives entre les données du jeu de données.
La recherche d'associations vise à construire un modèle basé sur des règles conditionnelles à partir d'une table de données constituée de valeurs d'attributs. Une règle conditionnelle se définie sous la forme suivante :

SI conditions **ALORS** résultats

où « conditions » et « résultats » sont des conjonctions d'expressions simples sur les attributs du jeu de données.
Par exemple, étant donné un jeu de données de transactions, où chaque ligne représente une liste d'articles achetés $(A, B, ..., Z)$ par un client, on peut extraire la règle d'association $A\ et\ B\ et\ C \rightarrow D\ et\ F$ interprétée par : les clients qui achètent les produits A, B et C tendent à acheter en même temps les produits D et F. Afin de ne générer que les relations significatives entre les ensembles d'articles, des mesures d'utilité (le $Support$) et de précision (la $Confiance$), empruntées aux statistiques, sont associées à chaque règle d'association. Les règles générées sont celles dont le $Support$ et la $Confiance$ sont supérieurs ou égaux à des seuils minimaux définis par l'utilisateur en fonction de ses objectifs et du type de données traitées.

Dans la suite, nous présentons la manière dont les problèmes des règles d'association sont en général formalisés [Agra93a, Agra93b]. Soit $I = \{i_1, i_2, ..., i_m\}$ un ensemble de m termes (ou items). Soit $B = \{t_1, t_2, ..., t_n\}$ un jeu de données de n transactions, chaque transaction t_i étant représentée par un sous-ensemble A de I appelé itemset et

identifié par un identifiant unique noté TID. Un sous-ensemble I de taille k est appelé un k-itemset. On dit qu'une transaction t_i « contient » un itemset A si et seulement si $A \subseteq t_i$.

Le *Support* d'un itemset A est le pourcentage de transactions de B qui contiennent A :

$$Support(I) = |\{t \in B; A \subseteq t\}|$$

Un itemset dont le *Support* est supérieur ou égal au seuil minimal de *Support* définit par l'utilisateur est appelé itemset fréquent. Une règle d'association est une implication de la forme $I_1 \Rightarrow I_2$ entre deux itemsets I_1, I_2 tels que $I_1 \cap I_2 = \emptyset$.

La $Confiance$ d'une règle d'association $r : I_1 \Rightarrow I_2$ est égale à la probabilité conditionnelle qu'une transaction contienne I_2 sachant qu'elle contient I_1 :

$$Confiance(r) = \frac{Support(I_1 \cup I_2)}{Support(I_1)}$$

Etant donné un jeu de données de transactions B, le problème de l'extraction des règles d'association dans B consiste à déterminer l'ensemble de règles d'association dont le *Support* et la $Confiance$ sont au moins égaux à des seuil minimaux de *Support* $support_{min}$ et de $Confiance$ $confiance_{min}$ définis par l'utilisateur.
L'extraction de règles d'association est un processus itératif et interactif [Fayy96a]. Le principe global de la génération de l'ensemble des règles d'association est le suivant : pour chaque itemset fréquent l_1 dans F (l'ensemble des itemsets fréquents) de taille supérieure ou égale à deux, tous les sous-ensembles l_2 de l_1 sont déterminés et la valeur du rapport $Support(I_1)/Support(I_2)$ est calculée. Si cette valeur est supérieure ou égale au seuil $confiance_{min}$ alors la règle d'association $I_2 \Rightarrow (I_2 - I_1)$ est générée. Les algorithmes d'extraction des itmesets fréquents génèrent pour chaque itération k un ensemble de k itemsets candidats. Les *Supports* de ces candidats sont calculés lors d'un seul et même balayage, ce qui permet de limiter le nombre total de balayages réalisés [Brin97, Baya98]. On cite parmi les algorithmes les plus connus, l'algorithme APRIORI [Agra94] qui réalise N itérations afin de déterminer tous les itemsets fréquents dans l'ordre croissant de leur taille, N étant la taille des plus grands itemsets fréquents. Afin de limiter le nombre de candidats considérés lors de chaque itération, en réduisant dynamiquement l'espace de recherche des itemsets fréquents, l'algorithme APRIORI se base sur les deux propriétés suivantes :

- *Tous les sous-ensembles d'un itemset fréquent sont fréquents.* Cette propriété permet de limiter le nombre de candidats de taille k générés lors de la $k^{ème}$ itération (parmi les 2^k itemsets de taille k existants) en réalisant une jointure conditionnelle des itemsets fréquents de taille $k-1$ découverts lors de l'itération précédente.

- *Tous les sur-ensembles d'un itemset infréquent sont infréquents.* Cette propriété permet de supprimer un candidat de taille k lorsque au moins un de ses sous-ensemble de taille $k-1$ ne fait pas partie des itemsets fréquents découverts lors de l'itération précédente.

Actuellement, les travaux sur les règles d'association visent à réduire l'ensemble de règles générées afin d'améliorer la pertinence du résultat.

Les avantages des règles d'association résident par exemple, dans le fait qu'elles exhibent des corrélations innattendues entre données a priori complètement indépendantes. De même, ces règles sont généralement aisées à interpréter. Les inconvénients résident au niveau du temps de recherche pour découvrir toutes les combinaisons possibles entre les données et dans l'extraction des règles selon un certain seuil de $Support$ et de $Confiance$.

2.2.2 Approche prédictive

Dans l'*apprentissage supervisé*, un expert du domaine donne les classes et fournit des exemples [1] de chaque classe, après avoir réalisé, par exemple, une classification préalable. Cette activité est aussi dénomée apprentissage à partir d'exemples. Le système doit découvrir un modèle permettant de déduire la classe d'une nouvelle donnée non étiquettée par une classe particulière. Elle est utile par exemple lors d'opérations de mailing pour cibler des clients potentiels et minimiser le nombre de non-réponses. De la même manière, cette technique peut être utilisée pour aider à déterminer si oui ou non un crédit peut être accordé à une personne en analysant les situations déjà rencontrées avec des personnes du même profil. Le but est de construire un modèle permettant de classer toute nouvelle donnée. Les travaux sur l'apprentissage supervisé ne sont pas spécifiques du domaine de la fouille de données ; dans le cadre de l'apprentissage automatique, on peut citer de nombreux résultats sur les algorithmes de classification [Gelf91, Zigh98, Kary99, Wang99, Zigh00].

Différentes techniques peuvent être mises en œuvre pour l'apprentissage supervisé des données, par exemple, l'induction d'arbres de décision ou la construction de réseaux de neurones [Crav94, Lu95]. La méthode la plus utilisée est celle consistant à construire des arbres de décision. Celle-ci est une des plus visuelles et donc des plus parlantes pour l'utilisateur.

[1] Les entrées d'un jeu de données sont également appelés « instances », « objets » ou « exemples ».

Les arbres de décision permettent de diviser des données en groupes basés sur les valeurs des variables. Ils sont des outils puissants et appréciés pour l'apprentissage supervisé et la prédiction. Cette technique est la plus utilisée et suscite le plus d'engouement car elle est plus simple et plus rapide que les réseaux de neurones et donne des modèles facilement compréhensibles par l'utilisateur.

Ainsi, les modèles de classifieur comme les arbres de décision s'expriment comme un ensemble de règles de classification de la forme :

SI description **ALORS** classe

On peut définir une classe unique ou des classes multiples. Un classe unique C, appelée description de caractéristiques, doit être construite ; tous les échantillons appartenant à cette classe sont des échantillons positifs. Le système doit alors construire une description pour ces derniers. Les échantillons négatifs sont les membres des autres classes. Par exemple, dans l'environnement animal, on peut définir une classe « poisson » et rechercher une description de caractéristiques, où tous les poissons servent d'exemples positifs et les autres animaux d'exemples négatifs.

On peut voir un arbre de décision comme un enchaînement hiérarchique de règles logiques construites de manière automatique à partir d'une base d'exemples. Un exemple est constitué d'une liste d'attributs, dont la valeur détermine l'appartenance à une classe donnée. La construction de l'arbre de décision consiste à utiliser les attributs, pour subdiviser progressivement l'ensemble d'exemples en sous-ensembles de plus en plus fins. Cette technique permet de déterminer les attributs significatifs pour un élément donné. Le mécanisme de base consiste à choisir un attribut comme racine et à développer l'arbre selon les attributs les plus significatifs. Il existe une grande variété d'algorithmes pour construire des arbres de décision, on cite par exemple CART (Classifixation And Regression Trees), CHAID (CHi-squared Automatic Interaction Detection) et C4.5. L'algorithme CART [Brei84] construit un arbre binaire en divisant les enregistrements au niveau de chaque nœud selon une fonction ne dépendant que du champ d'entrée, ensuite il procède en élaguant l'arbre. CHAID [Hart75], plutôt que de tenir compte de toutes les données puis d'élaguer ensuite ce qui est trop particulier, essaie d'arrêter la croissance de l'arbre avant la construction de branches qui devront être ensuite élaguées. C4.5 [Quin96], est un des algorithmes les plus connus et produit un nombre variable de branches par nœud alors que CART amène à un arbre binaire à chaque nœud.

Dans la suite du mémoire, nous utilisons à de multiples reprises, les résultats obtenus à l'aide d'un algorithme dérivé de l'algorithme d'induction d'arbres de décision C4.5. Ce dernier est lui même dérivé de l'algorithme ID3 « Induction Decision Tree » [Quin86] basé sur le gain informationnel comme critère de sélection des variables. Il s'agit de la méthode la plus « naïve » pour construire un arbre d'induction. L'algorihtme C4.5

représente en fait un ensemble d'algorithmes qui permettent d'une part de construire un arbre d'induction par le biais d'un critère appelé « le ratio du gain » et, d'autre part, d'élaguer les arbres, simplifier les règles et gérer les données manquantes. Les paragraphes suivants mettent l'accent sur la manière dont un arbre de décision est construit et dont l'élagage peut être réalisé.

Construction. Lors de la construction d'un arbre de décision, l'idée est de sélectionner l'attribut du jeu de données qui sépare au mieux celles-ci, afin de les répartir dans les nœuds fils. Il s'agit donc de définir par un critère mesurable ce qu'est une discrimination pour obtenir le meilleur partitionnement. Dans l'algorithme ID3, la mesure d'entropie est utilisée comme critère. Cette mesure fut introduite par C. E. Shannon et al. [Shan49] et mesure le « chaos » de l'information.

Soit $(p_1, p_2, ..., p_m)$ un élément du simplexe $\sum m$, il s'agit généralement d'un vecteur de probabilité. On appelle entropie de Shannon l'application h_s définie par :

$$\begin{aligned} h_s &: \sum m \longmapsto R_+ \\ h_s(p_1, p_2, ..., p_m) &= -\sum_{i=1}^{m} log_2(p_i) \end{aligned}$$

Les échantillons envisagés sont de taille finie, aussi, on estime ces probabilités par des fréquences empiriques. L'entropie est calculée pour chaque sommet d'un arbre. Soient N l'effectif total d'un jeu de données, N_k l'effectif des instances du sommet S_k et N_{ik} l'effectif des instances du sommet S_k qui appartiennent à la classe C_i. Avec la définition précédente, l'entropie d'un sommet S_k est donc :

$$h_s(S_k) = -\sum_{i=1}^{m} \left(\frac{f_{ik}}{f_k} \times log_2 \frac{f_{ik}}{f_k} \right)$$

avec $f_{ik} = \frac{n_{ik}}{N}$ et $f_k = \frac{n_k}{N}$.

Cette mesure permet d'indiquer la « qualité prédictive » d'un ensemble. Une propriété supplémentaire rend cette mesure extrêmement pratique : *l'entropie d'un ensemble de sous-ensembles est égale à la somme des entropies des sous-ensembles pondérées par leur rapport de taille à l'ensemble global.* Soit une partition S_k à P éléments, l'entropie $\mathcal{I}(S_P)$ de la partition S_P est égale à :

$$\begin{aligned}\mathcal{I}(S_P) &= \sum_{k=1}^{P}(f_k \times h(S_k)) \\ &= \sum_{k=1}^{P} f_k \times \left(-\sum_{i=1}^{m}\left(\frac{f_{ik}}{f_k} \times log_2\frac{f_{ik}}{f_k}\right)\right)\end{aligned}$$

Il est ainsi possible de mesurer l'entropie d'un nœud par rapport à une variable, mais également l'entropie de l'ensemble de ses nœuds fils. La différence entre ces deux mesures est appelée le *gain informationnel* qui représente le « pouvoir discriminatoire » de la classification obtenue. De fait, si on considère un nœud contenant un certain nombre d'instances du jeu de données, la paire attribut-valeur qui engendrera un ensemble de nœuds fils tel que le gain informationnel sera maximal, indiquera la meilleure discrimination possible sur ce nœud. La répétition de ce processus à partir de la racine de l'arbre et sur ses nœuds fils successifs et ce jusqu'à ce qu'aucune discrimination ne fournisse un gain informationnel positif, permet de construire l'arbre de décision correspondant au jeu de données. Les algorithmes de J. R. Quinlan sont conçus pour construire des arbres de décision n-aires. Plutôt que d'utiliser une paire attribut-valeur pour engendrer une bipartition, un attribut engendre une partition constituée d'autant d'ensembles que de valeurs discrètes de cet attribut. Pour des attributs ayant des ensembles importants de valeurs, cela provoque parfois des arbres trop étendus. Dans C4.5, J. R. Quinlan corrige ce problème par l'utilisation d'un critère de partitionnement différent, également basé sur l'entropie, mais qui pénalise les partitionnements trop importants. Une procédure d'élagage de l'arbre de décision après la fin de sa construction contribue également à réduire la taille des arbres produits.

Elagage. Il s'agit de réduire la taille de l'arbre après avoir développé celui-ci au maximum par le biais du processus décrit dans le paragraphe précédent. La réduction de l'arbre consiste alors à supprimer tous les sous-arbres qui ne vérifient pas une condition qui est basée sur le taux d'erreur et ce sur un jeu de données appelé « ensemble de test ». Il s'agit de mesurer si les feuilles issues d'un nœud père conduisent à un taux d'erreur moyen meilleur selon la règle suivante : si la moyenne des bornes supérieures des taux d'erreur sur les feuilles issues d'un nœud père est plus grande que la borne supérieure du taux d'erreur de ce nœud père, il faut alors supprimer les descendants du nœud père car l'erreur engendrée est en moyenne plus importante.

Limites. Il est admis que les algorithmes d'induction d'arbres de décision sont très performants. Leur complexité est fonction du nombre d'attributs discriminatoires sur les données en présence. L'induction d'arbres n-aires à partir de données discrètes se fait généralement en un temps raisonnable.

2.2.3 Autres tâches

Nous présentons ici les deux autres principales tâches qui sont les motifs séquentiels et la généralisation des données.

Motifs séquentiels

Ce problème fut introduit par R. Agrawal et al. dans [Agra95] puis repris et étendu dans [Srik96]. De nombreuses bases de données présentent une étiquette de temps associée à chaque entité. Il s'agit de mettre en évidence des séquences qui sont récurrentes. Par exemple si l'on considère une base de données de transactions de ventes d'articles, une date, un ensemble d'articles et un client sont associés à chaque transaction. Le problème consiste alors à rechercher les séquences d'articles qui sont récurrentes dans la base.

Généralisation

Il s'agit dans cette tâche de fournir un ensemble réduit de tuples qui décrivent les instances de la base de données à un niveau d'abstraction plus élevé. Il existe deux approches : l'approche par « data cube » [Gupt95, Hari96] et l'approche par induction orientée attributs [Mich83, Han93, Han96].
La première approche consiste à matérialiser les résultats de calculs coûteux qui sont fréquemment requis, plus particulièrement les calculs qui utilisent les fonctions d'agrégation (somme, moyenne, etc). Dans la seconde approche, la taxonomie des attributs de la base de données est utilisée afin de généraliser les instances et fusionner les doublons tout en maintenant leur décompte. Le résultat peut ensuite être utilisé afin, par exemple, de générer des règles d'association ou des séries chronologiques [Mann96].

2.3 Conclusion

Dans ce chapitre nous avons donné un aperçu général des techniques de fouille de données les plus utilisées. Dans la suite de ce mémoire, nous nous focalisons sur les règles de dépendance qui prennent la forme de règles de classification ou de règles d'association. Nous étudions dans le chapitre suivant les espaces de recherche des règles caractérisées par des mesures de qualité diverses. L'idée est d'étudier les propriétés de ces espaces et l'adéquation ou l'inadéquation des algorithmes de recherche proposés.

Chapitre 3

Règles et mesures de qualité

Sommaire

L'extraction automatique de connaissances à partir de données, comme nous l'avons mentionné dans le chapitre précédent, peut être définie comme la découverte de relations inattendues dans de très grands volumes de données dont la taille nécessite un traitement automatique. Dans le chapitre 2, nous avons présenté les diverses tâches de la fouille de données en nous focalisant sur celles qui sont à base de règles à savoir l'extraction de règles de classification et la recherche de règles d'association. Nous regroupons sous le qualificatif de *règles de dépendance* ces deux types de règles. Nous reprenons la distinction souvent faite entre modèle et motif. Par exemple, D. J. Hand et al. [Hand01] définissent un *modèle* relativement à une structure globale qui couvre toutes les données alors qu'un *motif* correspond à une description d'une partie de l'espace des données. De nombreuses applications de fouille de données sont plus spécialisées dans la recherche de motifs locaux que dans la recherche de modèles globaux. Dans le cadre de cette thèse, nous nous focalisons sur les modèles à base de règles de

la forme $A \rightarrow C$ où A et C sont des conjonctions de termes attribut-valeur $Att\ op\ a$ où Att est un attribut, a est une valeur et $op \in \{<, \leq, =, \geq, >\}$. Dans les applications de fouille de données, il est assez fréquent de rechercher des modèles à base de règles étant donné que ceux-ci sont aisément compréhensibles par l'utilisateur final et se révèlent utiles pour l'apprentissage de connaissances interprétables à partir des données. Un élément essentiel est de mesurer l'intérêt du lien de dépendance entre les prémisses A et le conséquent C des règles. Les algorithmes standards utilisent des mesures simples pour la sélection de règles. Mais pour satisfaire les objectifs spécifiques de la fouille de données, des indices plus spécifiques ont été proposés.

Les techniques traditionnelles des tâches que nous avons présentées dans le chapitre précédent, sont dirigées par les buts. En effet en classification supervisée, l'objectif est de trouver le meilleur classifieur, en classification il s'agit de trouver le meilleur ensemble de classes, etc. Un autre point de vue consiste à considérer la mesure de qualité des entités des modèles extraits comme un problème d'optimisation. Dans ce mémoire, nous assimilons *intérêt* à *qualité*. Ainsi, on peut considérer la fouille de données comme un problème d'optimisation, et par conséquent, il est nécessaire de mettre en œuvre des méthodes permettant de prendre en compte cet aspect. Dans ce cadre, nous considérons le concept de règle de dépendance qui recouvre règles de classification et d'association. Ces deux types de règles sont deux concepts importants en fouille de données. A. Freitas [Frei98a] met en évidence les différences majeures entre les deux concepts. Cependant, certains auteurs combinent les deux types de modèles. Par exemple, B. Liu et al. [Liu98] proposent d'intégrer les deux types de règles pour construire des classifieurs qui sont plus adaptés aux objectifs de la fouille de données. Cette approche utilise les techniques d'extraction des règles d'association pour la tâche de classement. L'algorithme fondateur APRIORI est adpaté pour cette tâche. Selon cette technique, les règles sont sélectionnées si elles satisfont des seuils minimaux de $Support$ et de $Confiance$ qui ne sont pas des critères standards pour les règles de classification. Ces règles prédisent une valeur d'attribut de classe et elles sont considérées individuellement. Dans [Dhar00], le système est conçu pour la recherche de règles de classification pour des applications financières. L'évaluation des règles est réalisée selon deux critères : le taux de mauvais classement sur des données inconnues pour le modèle entier, et les performances en termes de $Support$ et de $Confiance$ pour les règles prises individuellement. Il est en effet indiqué que dans les problèmes financiers, il est peu probable qu'un modèle puisse classer précisément tous les cas. Le système est conçu pour ne trouver éventuellement que peu de règles utiles et très précises. La qualité des règles est déterminée par le biais du $Support$, de la $Précision$, de l'$Entropie$ ou de certaines combinaisons de ces métriques sur des règles individuelles. Dans ce mémoire nous suivons une approche similaire : nous étudions l'intérêt et l'utilité de règles prises individuellement. Nous considérons les règles de manière générale comme des liens de dépendance entre attributs.

Intérêt des dépendances. Les critères simples pour la sélection de règles de qualité comme la *Précision* pour un modèle entier ou le *Support* et la *Confiance* pour des règles individuelles sont connus pour être insuffisants si l'on veut extraire des informations utiles et intéressantes. Pour la qualité globale d'un classifieur, et particulièrement pour les applications médicales ou la recherche d'informations, la *Sensiblité* et la *Spécificité* fournissent plus d'information que la qualité d'un modèle. L'utilité et l'intérêt peuvent avoir différents sens et on distingue des approches objectives et subjectives. Les critères subjectifs sont généralement basés sur une comparaison des règles apprises par rapport à une connaissance a priori sur les données. Une des premières mesures objectives, la mesure *Rule Interest* (RI) a été définie par G. Piatetsky-Shapiro et al. [PS91]. De nombreuses autres mesures ont été proposées pour l'évaluation de la qualité des informations extraites : le $Lift$ [IBM96], la $\mathcal{J}Mesure$ de R. M. Goodman et al. [Good91], la mesure définie par M. Sebag et al. [Seba88], la $Conviction$ [Brin97]. On peut citer encore le coefficient de corrélation ρ, la mesure de *Zhang* [Zhan00], le χ^2, le coefficient de *Loevinger* [Loev47], l'*indice d'implication* ou encore le *taux d'exemples et de contre-exemples*. Il a été observé [Hand01] que certaines d'entre elles sont relativement équivalentes. Cependant, il n'existe pas de définition de mesure de qualité unique. En effet, dans les travaux relatifs aux mesures d'intérêt, les auteurs fournissent généralement leur propre définition de l'intéréssabilité. Dans le cadre de cette thèse, nous ne considérons que les mesures objectives.

Comme nous le verrons dans la section 3.2, nous considérons dans ce mémoire deux types de règles : d'une part les règles *simples* qui sont constituées d'une seule condition dans leur conséquent, d'autre part, les règles *généralisées* qui ont plusieurs conditions dans leur conséquent. Notre objectif est d'étudier les espaces de recherche des mesures associés à ces deux types de règles sur divers jeux de données. Aussi, ce chapitre présente dans un premier temps un certain nombre de mesures ainsi que plusieurs travaux relatifs à ces mesures. Enfin, nous étudions leurs espaces de recherche et l'intérêt de combiner plusieurs de ces mesures pour la recherche de règles intéressantes.

3.1 Sélection des règles

Cette section présente dans un premier temps les mesures sur la base desquelles nous avons mené notre approche expérimentale. Nous fournissons leurs définitions. Dans un second temps, nous présentons un état de l'art de certaines études disponibles dans la littérature sur les mesures.

3.1.1 Mesures de qualité

Nous présentons ici certaines mesures de qualité. Celles-ci sont les plus fréquemment utilisées en fouille de données et sont destinées à mesurer la qualité de motifs exprimés sous forme de règles de dépendance. Nous précisons les définitions de ces mesures.

Nous utilisons la notion de « couverture d'un ensemble A par un ensemble C » qui peut être définie comme étant la quantité d'instances de A appartenant également à C.

Nous définissons les notions de *positif* et *négatif* pour une règle $r : A \rightarrow C$ comme étant respectivement une instance vérifiant A et une instance ne vérifiant pas A. Une instance du jeu de données est dite « étiquetée » par la valeur de C si cette instance présente cette valeur pour l'attribut C. Dans ce cas nous notons :

- V_p (vrai positif) : le nombre d'instances du jeu de données **satisfaisant** A et **étiquetées** par la valeur de C.
- F_n (faux négatif) : le nombre d'instances du jeu de données **ne satisfaisant pas** A et **étiquetées** par la valeur de C.
- F_p (faux positif) : le nombre d'instances du jeu de données **satisfaisant** A et **non étiquetées** par la valeur de C.
- V_n (vrai négatif) : le nombre d'instances du jeu de données **ne satisfaisant pas** A et **non étiquetées** par la valeur de C.

Sensibilité

Nous adaptons la définition de *Sensibilité* à une règle. La *Sensibilité* d'une règle $r : A \rightarrow C$, que l'on note $Sensibilité(r)$, permet de mesurer la proportion d'instances du jeu de données qui sont classées dans C par la règle r parmi toutes les instances effectivement étiquetées par C. Elle évalue la couverture de C par A. Cette mesure est définie de la façon suivante :

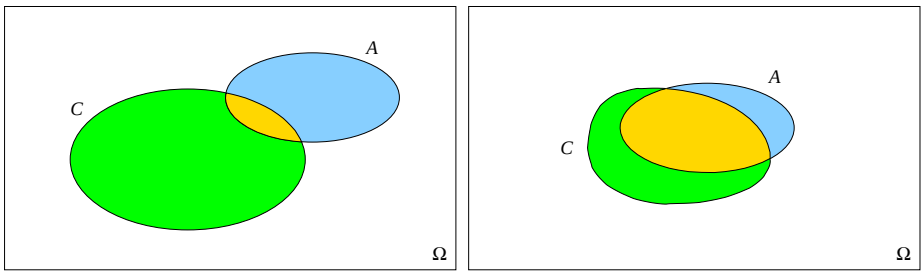
$$\boxed{Sensibilité(r) = \frac{V_p}{V_p + F_n} = p(A/C)} \tag{3.1}$$

Les diagrammes de Venn de la figure 3.1 illustrent l'influence de l'optimisation de cette mesure sur les caractéristiques de l'antécédent A et du conséquent C des règles.

Spécificité

La $Spécificité$ d'une règle $r : A \rightarrow C$, que l'on note $Spécificité(r)$, permet de mesurer la proportion d'instances du jeu de données qui ne sont pas classées dans C par la règle r parmi toutes les instances effectivement non étiquetées par C. Elle évalue la couverture de A par C. Cette mesure est définie de la façon suivante :

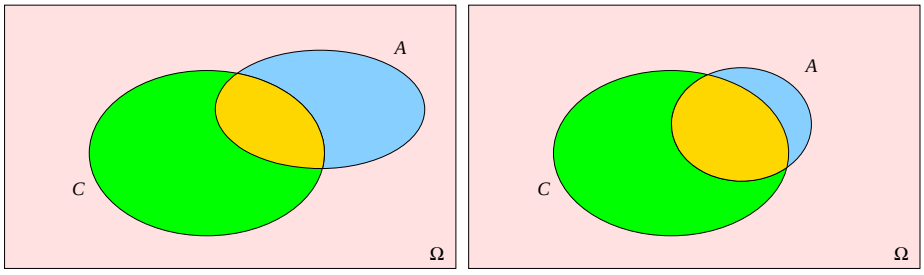
$$\boxed{Spécificité(r) = \frac{V_n}{V_n + F_p} = p(\neg A/\neg C) = 1 - p(A/\neg C)} \tag{3.2}$$



(a) (b)

FIG. 3.1 – *Diagrammes de Venn de la Sensibilité. Optimiser la Sensibilité revient à augmenter la couverture de C par A et à diminuer C, ce qui est traduit par le passage de (a) à (b).*

Les diagrammes de Venn de la figure 3.2 illustrent l'influence de la maximisation de cette mesure sur les caractéristiques de l'antécédent A et du conséquent C des règles.



(a) (b)

FIG. 3.2 – *Diagrammes de Venn de la Spécificité. Optimiser la Spécificité revient à augmenter la surface rose (représentant $\neg(A \cup C)$), ce qui est traduit par le passage de (a) à (b).*

La *Sensibilité* et la *Spécificité* sont définies à l'origine pour un classifieur dans le but d'apporter une idée plus précise sur ses performances. Ces deux mesures sont particulièrement utiles lorsque la distribution des classes est inégale.

Support

Le $Support$ et la $Confiance$ sont les mesures traditionnellement associées aux règles d'association. Le $Support$ d'une conjonction de termes $A \wedge C$, noté $Support(A \cap C)$ permet de mesurer la proportion d'instances du jeu de données qui satisfont simultanément l'antécédent A et le conséquent C de la règle $r : A \rightarrow C$. Il est toutefois d'usage d'exprimer le $Support$ en fonction de la règle r et ce de la façon suivante :

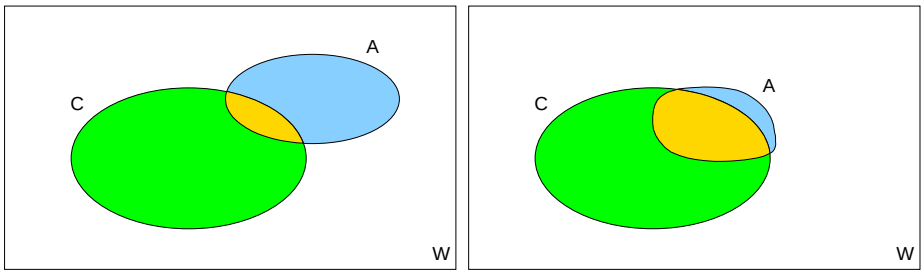
$$\boxed{Support(r) = p(A \cap C)} \tag{3.3}$$

Confiance

La $Confiance$ [Agra93b] d'une règle $r : A \rightarrow C$, que l'on note $Confiance(r)$, permet de mesurer la proportion d'instances du jeu de données qui satisfont simultanément l'antécédent A et le conséquent C de la règle r sachant que l'antécédent est vérifié. Cette mesure est définie de la façon suivante :

$$\boxed{Confiance(r) = \frac{Support(r)}{Support(A)} = p(C/A)} \tag{3.4}$$

Les diagrammes de Venn de la figure 3.3 illustrent l'influence de la maximisation de cette mesure sur les caractéristiques de l'antécédent A et du conséquent C des règles.



(a) (b)

FIG. 3.3 – *Diagrammes de Venn de la Confiance. Optimiser la Confiance revient à augmenter la couverture de A par C et à diminuer A, ce qui est traduit par le passage de (a) à (b).*

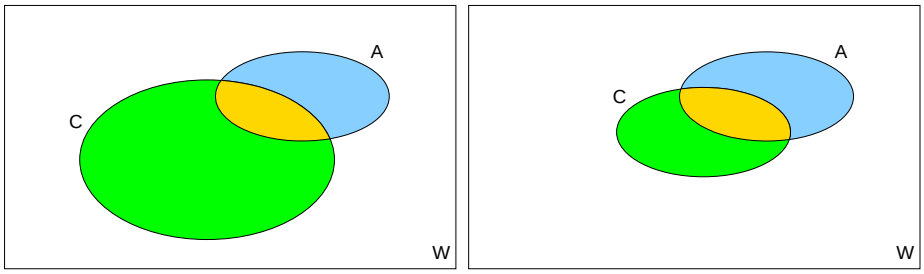
La faiblesse de la $Confiance$ est de ne pas prendre en compte $p(C)$. En effet, si l'on considère A « petit » et C « grand » et tels que $A \in C$, alors $|A \cap C| = |A|$ et la $Confiance$ est maximale. Cependant, il n'y a pas de relation de cause à effet entre A et C. Les deux mesures suivantes, le $Lift$ et la mesure $Rule\ Interest$, corrigent ce défaut.

Lift

Le $Lift$ [IBM96] d'une règle $r : A \rightarrow C$, noté $Lift(r)$, permet de mesurer la différence avec la situation d'indépendance entre A et C. En d'autres termes, elle permet de mesurer la différence entre la connaissance *a priori* sur C et la connaissance *a posteriori* sur C. Cette mesure est définie de la façon suivante :

$$\boxed{Lift(r) = \frac{Confiance(r)}{p(C)} = \frac{p(C/A)}{p(C)}} \tag{3.5}$$

Cette mesure est symétrique et de fait, mesure la co-occurence de A et C. Les diagrammes de Venn de la figure 3.4 illustrent l'influence de l'optimisation de cette mesure sur les caractéristiques de l'antécédent A et du conséquent C des règles.



(a) (b)

FIG. 3.4 – *Diagrammes de Venn du $Lift$. Optimiser le $Lift$ revient à augmenter le ratio de A par C par rapport à la couverture de tout l'espace par C, ce qui est traduit par le passage de (a) à (b) (les intersections de A et C dans les deux diagrammes sont égales).*

Rule Interest

La mesure *Rule Interest* [PS91] d'une règle $r : A \rightarrow C$, notée $RI(r)$, permet de mesurer tout comme le $Lift$ défini précédemment, la différence avec la situation d'indépendance entre A et C c'est-à-dire, la différence entre la connaissance *a priori* sur C et la connaissance *a posteriori* sur C. Cette mesure est définie de la façon suivante :

$$\boxed{RI(r) = |A| \times (Confiance(r) - p(C))} \tag{3.6}$$

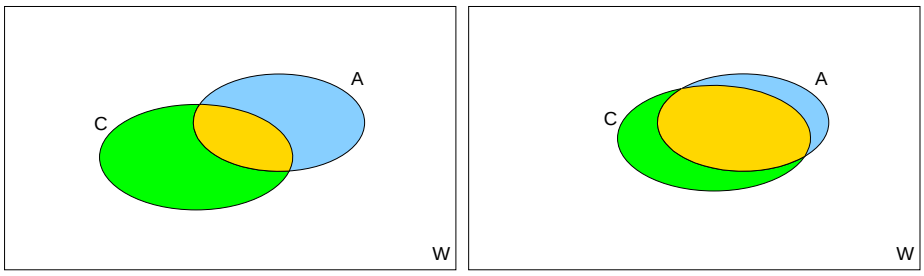
Cette mesure est symétrique comme le $Lift$. Cependant, elle prend en compte le *Support* de l'antécédent de la règle afin de mesurer la généralité de celle-ci.

Taux de succès

Le *Taux de succès* noté Ts, est principalement utilisé pour l'évaluation d'un classifieur mais on peut le définir pour une règle $r : A \rightarrow C$ de la façon suivante :

$$\boxed{Ts(r) = p(\neg A \cap \neg C) + p(A \cap C)} \tag{3.7}$$

Les diagrammes de Venn de la figure 3.5 illustrent l'influence de la maximisation de cette mesure sur les caractéristiques de l'antécédent A et du conséquent C des règles.



(a) (b)

FIG. 3.5 – *Diagrammes de Venn du Taux de succès. Optimiser le Taux de succès revient a augmenter l'intersection des deux ensembles, ce qui est traduit par le passage de (a) à (b).*

$\mathcal{J}$Mesure

La $\mathcal{J}Mesure$ [Good91] d'une règle $r : A \rightarrow C$, notée $\mathcal{J}Mesure(A, C)$ permet de mesurer d'une certaine manière la quantité d'information contenue dans la règle. Cette mesure est définie de la façon suivante :

$$\boxed{\mathcal{J}Mesure(A, C) = p(A) \times (\mathcal{J}Mesure_1(A, C) + \mathcal{J}Mesure_2(A, C))} \tag{3.8}$$

avec :

$$\mathcal{J}Mesure_1(A, C) = p(C/A) \times log_2 \left(\frac{p(C/A)}{p(C)} \right)$$

et

$$\mathcal{J}Mesure_2(A, C) = (1 - p(C/A)) \times log_2 \left(\frac{1 - p(C/A)}{1 - p(C)} \right)$$

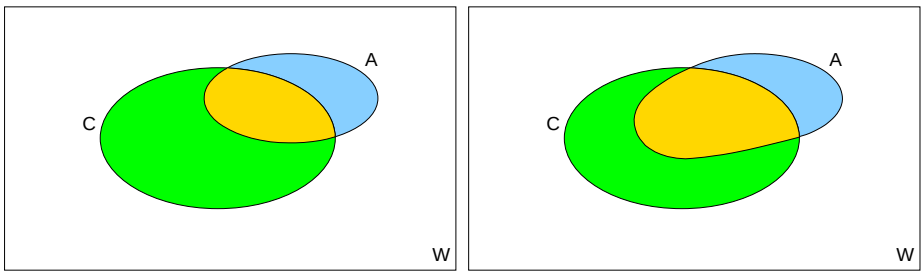
Le premier terme $\mathcal{J}Mesure_1(A, C)$ mesure le ratio de connaissance *a posteriori* sur C par rapport à A et la connaissance *a priori* sur C comme le $Lift$. Cependant, le second terme, $\mathcal{J}Mesure_2(A, C)$ introduit un biais favorisant l'émergence de règles contradictoires dans les modèles, c'est pourquoi l'auteur dans [Arau99] n'utilise que le premier terme $\mathcal{J}Mesure_1(A, C)$ de la $\mathcal{J}$Mesure.

Coefficient de Sebag

Le coefficient de *Sebag* [Seba88] d'une règle $r : A \rightarrow C$, noté $Sebag(r)$, permet de mesurer le ratio de vrais positifs par rapport aux vrais négatifs du jeu de données. Cette mesure est définie de la façon suivante :

$$\boxed{Sebag(r) = \frac{p(A \cap C)}{p(A \cap \neg C)}} \tag{3.9}$$

Les diagrammes de Venn de la figure 3.6 illustrent l'influence de la maximisation de cette mesure sur les caractéristiques de l'antécédent A et du conséquent C des règles.



(a) (b)

FIG. 3.6 – *Diagrammes de Venn du coefficient de Sebag. Optimiser le coefficient de Sebag revient à augmenter d'une part la couverture de $\neg C$ par A ou celle de C par A, ce qui est traduit par le passage de (a) à (b).*

Conviction.

La *Conviction* [Brin97] d'une règle $r : A \rightarrow C$, notée $Conviction(r)$, permet de mesurer l'implication logique $A \rightarrow C$ en évaluant le degré de dépendance entre A et $\neg C$. Cette mesure est définie de la façon suivante :

$$\boxed{Conviction(r) = \frac{p(A)}{p(A/\neg C)}} \tag{3.10}$$

Cette mesure n'est pas symétrique contrairement au $Lift$ et à la mesure *Rule Interest*.
Les diagrammes de Venn de la figure 3.7 illustrent l'influence de la maximisation de cette mesure sur les caractéristiques de l'antécédent A et du conséquent C des règles.

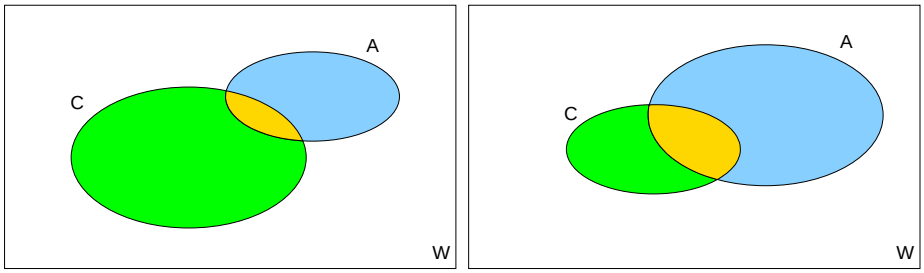


FIG. 3.7 – *Diagrammes de Venn de la Conviction. Optimiser la Conviction revient à augmenter l'intersection des deux ensembles A et C et à diminuer la cardinalité de C, ce qui est traduit par le passage de (a) à (b).*

Autres mesures

Le coefficient de corrélation linéaire ρ est une mesure de liaison linéaire entre deux variables quantitatives. Cette mesure est définie de la façon suivante :

$$\rho(r) = \frac{p(A \cap C) - p(A) \times p(C)}{\sqrt{p(A) \times p(\neg A) \times p(C) \times p(\neg C)}}$$

Le coefficient de $Zhang$ [Zhan00]. Cette mesure est définie de la façon suivante :

$$Zhang(r) = \frac{p(A \cap C) - p(A) \times p(C)}{max\{p(A \cap C) \times p(\neg C), p(C) \times (p(A) - p(A \cap C))\}}$$

Le χ^2 est un outil statistique qui mesure l'indépendance de deux variables. Cette mesure est définie de la façon suivante :

$$\chi^2 = n \times \frac{(p(A \cap C) - p(A) \times p(C))^2}{p(A) \times p(\neg A) \times p(C) \times p(\neg C)}$$

Le coefficient de *Loevinger* [Loev47] permet de mesurer l'écart à la situation d'indépendance. Cette mesure est définie de la façon suivante :

$$Loevinger(r) = 1 - \frac{p(A \cap \neg C)}{p(A) \times p(\neg C)}$$

L'indice d'implication $IndImp$ [Guil00] permet d'éliminer les règles non significatives c'est-à-dire les règles dont le conséquent est vérifié par la majorité des instances du jeu de données. Cette mesure est définie de la façon suivante :

$$IndImp(r) = \sqrt{N} \times \frac{p(A \cap \neg C) - p(A) \times p(\neg C)}{\sqrt{p(A) \times p(\neg C)}}$$

avec N le nombre d'instances du jeu de données.

Le taux d'exemples et de contre-exemples TEC. Cette mesure est définie de la façon suivante :

$$TEC(r) = \frac{p(A \cap C) - p(A \cap \neg C)}{p(A \cap C)}$$

L'intensité d'implication

S. Guillaume [Guil] inscrit ses travaux dans le contexte de la mesure des règles d'association. Aider l'utilisateur dans sa recherche de règles intéressantes par l'utilisation de mesures pertinentes lui fournissant un pré-classement sur l'intérêt des règles est l'objectif de ses travaux. Pour cela, l'auteur associe deux mesures à un algorithme d'extraction automatique de règles d'association. La première utilisée est l'intensité d'implication qui est une mesure statistique et la seconde est une mesure probabilistique. Le *Support* est toujours utilisé afin de filtrer les règles lorsque le jeu de données est volumineux. Cependant, ces mesures concernent les variables logiques et nécessitent pour les autres types de variables un codage approprié pouvant perturber la pertinence des résultats. De plus, la complexité des algorithmes d'extraction de règles d'association est généralement fonction du nombre d'attributs, le codage peut se heurter à une explosion combinatoire ou bien engendrer un nombre non exploitable de règles dont la plupart sont redondantes et peu significatives. L'auteur fournit alors une mesure implicative pour pallier ces problèmes : il s'agit de l'intensité d'implication ordinale, appliquable directement sur les variables numériques ainsi que sur les variables quantitatives ordinales à l'issue d'un codage approprié de celles-ci dans l'ensemble des réels. Cette mesure qui est une généralisation de l'intensité d'implication et de l'intensité de propension, permet d'avoir une démarche descendante dans l'analyse des règles et de faciliter ainsi son analyse. Un premier ensemble de règles beaucoup moins important et portant sur les variables dans leur globalité est d'abord obtenu, cet ensemble représente le comportement global de la population. Dans un second temps, il est possible de descendre au plus près de la population par un affinage des règles afin d'étudier le comportement des sous-populations, dans le but d'aider l'utilisateur dans la recherche de règles les plus intéressantes.

3.1.2 Un cas particulier : les règles rares

Nous utilisons le terme de « règle rare » pour traduire l'expression anglaise de « small disjunct » ou « rare event » ; une règle est dite « rare » si celle-ci couvre peu d'instances du jeu de données. Le problème lié à ce concept est que les règles rares ont paradoxalement un taux d'erreur plus grand que des règles plus générales. Le taux d'erreur représente le nombre d'instances du jeu de données qui sont mal classées par la règle. Cette mesure concerne à l'origine les règles « générales ». Ce phénomène fait des règles rares un domaine d'intérêt, étant donné qu'elles ont un impact plus large sur l'efficacité globale du concept appris. A titre d'exemple, sur le jeu de données « Vote » que nous présentons plus loin, les règles rares couvrent collectivement 20% des instances de l'ensemble test soumis à l'algorithme d'induction d'arbres de décision C4.5 et sont responsable de 90% des erreurs de classification [Weis00a].

Les travaux relatifs aux règles rares font état de plusieurs terminologies. Dans le cadre de ce chapitre, nous avons adopté une terminologie qui nous est propre. Les modèles que nous considérons sont des ensembles de règles [1], une règle permettant de caractériser un ensemble d'instances d'un jeu de données. La couverture [2] d'une règle est la proportion d'exemples du jeu de données qui vérifient à la fois l'antécédent et le conséquent de la règle. Une règle de forte couverture est dénommée règle globale [3] et une règle ayant une faible couverture est une règle rare [4]. Bien entendu, ces notions sont dépendantes d'un seuil à partir duquel une couverture est qualifiée de « faible ». Les règles rares et intéressantes selon un ou plusieurs autres critères sont appelées pépites de connaissance. Ces règles sont intéressantes dans un contexte qui est défini par l'utilisateur. Plusieurs travaux sont relatifs aux événements rares [Holt89, Quin91, Ali92, Dany93, Weis01].

Voici un résumé chronologique des travaux relatifs aux règles rares.

Dans [Holt89] les auteurs décrivent et définissent le problème des règles rares. Ils démontrent que celles-ci sont présentes dans de nombreux domaines et particulièrement dans deux d'entre eux où les règles rares sont une plus grande source d'erreurs que les règles générales. Ils décrivent des stratégies pour améliorer l'apprentissage en présence de règles rares. Il est démontré que la stratégie consistant à éliminer toutes les règles rares est inefficace. Le biais favorisant la généralité utilisée dans les algorithmes comme ID3 est adapté à la découverte de règles générales mais il ne l'est pas pour les règles rares. Pour illustrer cela, ils utilisent un biais favorisant la généralité maximale pour la découverte de règles générales et un biais plus spécifique pour la découverte de règles rares.

[1] Disjunct en anglais.
[2] Disjunct size en anglais.
[3] Large disjunct en anglais.
[4] Small disjunct en anglais.

Dans [Quin91] l'auteur décrit une méthode pour améliorer l'efficacité des règles rares en prenant en compte en plus du taux d'erreur, la distribution de la classe. En effet, la prise en compte du taux d'erreur est une approche naïve et peut se révéler inefficace en raison de son manque de signification statistique. Il s'agit d'obtenir des règles rares qui prédisent la classe la plus présente dans le jeu de données. L'auteur utilise ce concept dans l'estimation du taux d'erreur et montre au travers d'expérimentations, que l'utilisation de ce taux d'erreur *étendu* surpasse la méthode naïve, précisément quand les règles rares prédisent la classe minoritaire.

Dans [Ali92], les auteurs décrivent comment l'algorithme HYDRA, réduit le problème lié aux règles rares. Cet algorithme détermine des règles qu'il apprend grâce à une mesure de fiabilité de celles-ci. Cette approche assigne une faible fiabilité aux règles rares, ce qui diminue d'autant leur impact.

Dans [Dany93], les auteurs mettent en lumière que dans le domaine des télécommunications, les règles rares sont nécessaires bien qu'ayant un taux d'erreur relativement élevé. Ils montrent qu'en utilisant un intervalle des couvertures pour les valeurs de règles rares, il est possible d'utiliser l'apprentissage par induction pour constituer un modèle de règles pour les problèmes de diagnostique des réseaux locaux de télécommunication. Cependant, lorsque les données sont bruitées, il n'est pas possible de faire un tel apprentissage pour deux raisons : en premier lieu il est difficile de faire la distinction entre les données bruitées et les vraies exceptions et deuxièmement, les erreurs dans ce domaine se produisent plutôt de façon systématique que de façon aléatoire.

Dans [Ting94], l'auteur décrit une méthode pour améliorer les performances des règles rares en utilisant un biais spécifique maximum. Cependant, cette méthode n'affecte pas les performances des règles rares comme c'est le cas dans [Holt89]. La méthode utilise l'algorithme d'induction d'arbres de décision C4.5 pour déterminer si une instance est couverte par une règle rare ou par une règle générale. Si celle-ci est couverte par une règle générale, alors C4.5 est utlisé pour classer l'instance sinon, l'algorithme IB1 est mis en œuvre pour le classer.

Empiriquement on a pu observer qu'un concept appris était composé de quelques règles globales et de règles locales. Les règles locales sont intéressantes dans la mesure où elles améliorent la précision d'un concept en représentant des événements exceptionnels. Cependant elles peuvent tout aussi bien représenter du bruit et G. M. Weiss prouve dans ces études qu'elles sont à l'origine d'importants taux d'erreur. En analysant des données médicales il a pu mettre en évidence que 0,1% des règles locales représentaient 80% des erreurs de classification. Dans [Weis95], l'auteur met en lumière l'interaction entre le bruit, les événements rares et les règles rares. Des jeux de données synthétiques sont exclusivement utilisés afin de pouvoir construire des événements

rares. Ce travail démontre que le bruit aléatoire dans les jeux de données, le bruit dans les classes et les valeurs manquantes impliquent l'augmentation du taux d'erreur des règles rares par rapport à celui des règles générales. Une explication de ce comportement est donnée. Dans [Weis98], les auteurs étendent le travail précédent en examinant l'impact du bruit sur les règles rares dans deux domaines réels. En particulier, ce travail a été initié par le travail de A. P. Danyluk et F. J. Provost [Dany93] relatif à la difficulté de distinguer le bruit des événements rares. G. M. Weiss montre de façon empirique que les classes bruitées augmentent le nombre de règles rares et également le taux d'erreur de celles-ci. Enfin dans [Weis00b], les auteurs fournissent l'analyse la plus étendue à ce jour relatives aux règles rares. Elle porte sur un ensemble de trente jeux de données. Il introduit une mesure appelée la *Concentration d'erreur*, qui résume à quel point les erreurs sont concentrées sur les règles rares. En effet, ce phénomène bien que souvent observable n'est pas présent dans tous les domaines. G. M. Weiss déduit de ces études expérimentales une classification d'un ensemble de jeux de données dont nous nous sommes servis au cours de nos travaux. L'approche de G. M. Weiss des règles locales est cependant très différente de la notre. Il essaye de comprendre leur comportement afin de résoudre le problème des erreurs contenues dans les classifieurs, tandis que notre travail s'est plutôt orienté sur la façon d'extraire les meilleures règles locales possibles. Ce travail étudie de même la relation entre les règles rares, la taille de l'ensemble d'apprentissage soumis à l'algorithme C4.5 et l'élagage utilisé dans ce dernier. Les auteurs concluent que l'élagage n'est pas efficace quand les erreurs ne sont pas concentrées sur les règles rares ; de même, ils établissent que l'augmentation de la taille de l'ensemble d'apprentissage augmente la concentration d'erreurs sur les règles rares. Enfin, ils concluent que la définition de règles rares peut dépendre de la taille de l'ensemble d'apprentissage.

Notre approche est différente puisqu'elle est relative à la découverte de règles rares et intéressantes. Nous définissons les règles rares de la manière suivante : soit une règle généralisée de la forme $r : A \rightarrow C$, r est dite rare ssi :

- Elle couvre peu d'exemples du jeu de données, c'est-à-dire $|A \cap C|$ est inférieur à un seuil fixé.

- Le nombre d'instances couvertes par A et $\neg C$ doit être faible.
  L'antécédent de la règle doit satisfaire le moins d'instances possible appartenant à $\neg C$. Autrement dit, la règle doit avoir un faible taux d'erreur.

Formellement, une règle $r : A \rightarrow C$ est dite rare ssi étant donnés les seuils δ et ϵ fixés par l'utilisateur :

$$\frac{|A \cap C|}{N} < \delta \text{ et } \frac{|A \cap \neg C|}{N} < \epsilon$$

N étant le nombre d'instances du jeu de données.

Cette définition nous conduit à considérer deux cas. Dans les deux cas, la taille de A est faible :

- C est « petit » : A recouvre alors presque totalement C. Ce cas est illustré dans la figure 3.8 (a).

- C est « grand » : ce cas peut se produire lorsque quelques règles globales recouvrent presque totalement une classe C, et plusieurs règles locales permettent d'affiner la couverture de C. Ce cas est illustré dans la figure 3.8 (b).

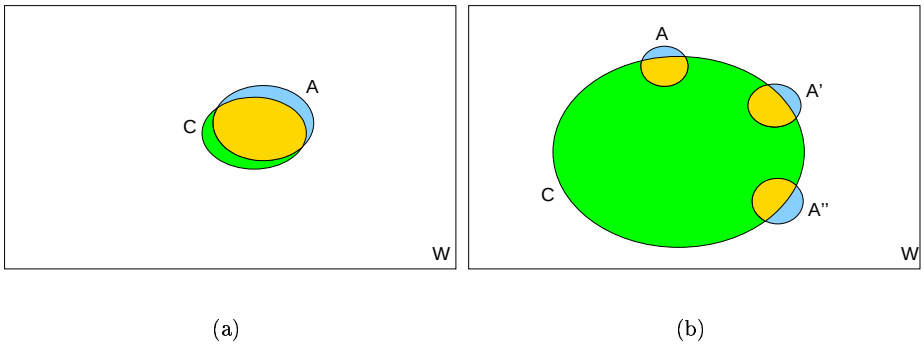


FIG. 3.8 – *Règles rares. (a) : cas où C est « petit », A recouvre presque totalement C. (b) : cas où C est grand, plusieurs règles locales permettent d'affiner la couverture de C.*

Dans la suite, nous étudions les corrélations de certaines mesures prises deux à deux et nous montrons la nécessité de recourir à des méthodes spécifiques d'optimisation multi-critères. Nous envisageons également le cas de trois mesures, attendu qu'il est possible de considérer un nombre plus élevé de mesures.

3.1.3 Travaux relatifs aux mesures

Il existe de nombreuses mesures qui ont été introduites au fil du temps. Cependant, il n'est pas aisé de distinguer leurs points communs ainsi que leurs différences. C'est dans cette optique que certains travaux ont été réalisés. Il s'agit dans cette section de présenter ceux qui nous paraissent être les plus significatifs dans le contexte de la fouille de données pour l'évaluation de motifs.

Dans le cadre de cette thèse, nous nous intéressons principalement aux mesures objectives. Cependant, il est intéressant de donner un aperçu de ce que sont les mesures subjectives. Ces mesures, également dénommées « mesures guidées par l'utilisateur » sont peu étudiées actuellement. Elles sont principalement basées sur la prise en compte de connaissances déjà acquises par l'utilisateur. Ainsi, lorsque le processus de fouille de données découvre un modèle, celui-ci est considéré comme étant intéressant s'il est nouveau ou surprenant pour l'utilisateur, c'est-à-dire s'il contredit certaines des connaissances fournies au préalable par l'utilisateur. B. Liu [Liu99, Liu01] utilise dans ses travaux le concept d'impression générale afin de sélectionner les règles intéressantes. Il s'agit pour l'utilisateur de spécifier ses impressions générales concernant les relations entre les données dans le domaine d'application. Ces impressions générales sont fournies par le biais de règles de la forme *SI-ALORS*. Il ne s'agit pas de proposer forcément des règles précises, car souvent l'utilisateur n'est pas en mesure de les fournir. Cette notion d'impression générale repose donc sur des règles assez générales mais non-triviales afin de fournir des indices utiles au processus de fouille de données pour extraire des règles intéressantes.
Nous avons rassemblé ci-dessous les travaux les plus significatifs relatifs aux mesures de qualité. Nous les regroupons en deux catégories : les études analytiques de mesures et les études des espaces de recherche des règles.

Etudes analytiques de mesures

R. J. Hilderman et al., dans leur travail sur les connaissances obtenues par généralisation ou agrégation [Hild01], ont été confrontés au choix de la mesure à employer pour supprimer les règles inutiles. Afin de sélectionner les règles intéressantes ils ont choisi de comparer une dizaine de mesures de qualité. Bien que cette approche s'applique à un domaine particulier, ils ont réfléchi sur ce que pouvait être une bonne mesure d'intérêt en termes formels et génériques. Plusieurs principes ont donc été définis afin de repérer les meilleures mesures, celles les plus à même de classer de façon satisfaisante les règles générées. Ils définissent comme ensemble de propriétés nécessairement vérifiées par une mesure d'intérêt qu'ils nomment « théorie de l'intéressabilité » :

- Les valeurs de mesures doivent être bornées.
- Une propriété de commutativité ne donnant pas d'importance à l'ordre des attributs caractérisant un exemple.

La démarche de R. J. Hilderman et al. est intéressante, mais elle est consacrée aux agrégations et généralisations.

S. Lallich et al. [Lall02] se focalisent plutôt sur la recherche de règles d'association les plus intéressantes. Les algorithmes les plus utilisés pour cette tâche sont du type APRIORI et sont fondés sur le *Support* et la $Confiance$. Les auteurs partent du

constat selon lequel ces algorithmes ignorent des règles intéressantes et que certaines règles n'ont pas d'intérêt. En effet, le processus d'extraction écarte les règles ayant un petit *Support* alors que certaines peuvent avoir une très forte $Confiance$ et présenter un certain intérêt. Toutefois, si on abaisse le seuil de *Support* on obtient une grande quantité de règles ce qui rend très difficile la tâche de sélection. De plus d'après les auteurs, les seules conditions de *Support* et de $Confiance$ ne suffisent pas à assurer le réel intérêt d'une règle. Un nombre important de mesures étant apparues pour évaluer la qualité d'une règle de manière plus précise, S. Lallich propose alors un ensemble de critères permettant d'évaluer les mesures à leur tour. Voici quelques-uns d'entre eux :

- Il est souvent nécessaire que la mesure ait un sens concret, qu'elle puisse parler à l'utilisateur, qui n'est peut-être pas expert du domaine.

- Une mesure doit permettre de distinguer les différentes règles faisant intervenir A et C. Une mesure dissymétrique est préférable à une mesure qui va évaluer de la même façon $A \rightarrow C$ et $C \rightarrow A$.

- Une règle peut être inattendue si elle possède, par exemple, peu de contre-exemples ; les mesures les prenant en compte sont intéressantes. Une règle est dite inattendue si elle diffère des connaissances a priori que l'expert possède sur le domaine.

- Les mesures peuvent être plus ou moins sensibles à la dilatation des données ; en présence de jeux de données de tailles extrêmes la mesure peut prendre alors des valeurs difficilement compréhensibles.

- Il est souhaitable qu'une mesure varie de façon non linéaire en fonction de l'apparition des exemples ou des contre-exemples, variation lente au début pour tenir compte du bruit, plus rapide ensuite.

L'étude présentée évalue les différentes mesures par rapport à chacun des critères énoncés et propose une analyse formelle de certaines mesures. On peut se poser la question concernant d'autres critères, comme celui de transitivité.

P. Lenca et al. [Lenc02] ont eu une approche similaire en proposant une technique d'aide à la décision multi-critères pour faire le choix d'une mesure adéquate aux attentes de l'utilisateur. Le choix de la bonne mesure reste un problème largement ouvert qui résulte pour les auteurs de différents points :

- Il est difficile de comparer des mesures de nature ou de valeur différentes.

- Il est difficile d'exprimer certains critères au moyen d'une mesure.

- Il est diffcile de retranscrire au travers de ces mesures les véritables critères des utilisateurs qui sont assez subjectifs.

Les auteurs définissent quatre sous-problèmes :

- Choisir une mesure ou un sous-groupe restreint de mesures.
- Classer les mesures de la meilleure à la moins bonne.
- Décrire les mesures et/ou leurs conséquences de façon normalisée.

Les auteurs se sont penchés sur ce dernier point, et ont pour un ensemble de mesures et de critères, établi une matrice de décision attribuant pour chaque mesure une note pour chaque critère de choix. Un critère de choix étant une fonction de l'ensemble des mesures étudiées dans un ensemble ordonné permettant de mesurer les préférences d'un décideur. Enfin les critères et préférences de l'utilisateur, permettent avec l'aide des matrices de décision de définir les mesures les plus adaptées à une situation donnée.

A. Freitas [Frei99b] présente différents facteurs influençant l'évaluation du degré d'intérêt des règles découvertes par un algorithme de fouille de données. L'auteur met l'accent sur plusieurs facteurs relatifs à l'intérêt des règles qui sont souvent négligés dans la littérature. D'autre part, il fournit une méthode pour prendre en compte ces facteurs dans les mesures existantes. Enfin, il introduit un nouveau critère pour mesurer la notion de surprise d'une règle qui est liée à la présence simultanée de plusieurs attributs dans l'antécédent de la règle. En effet, si l'on considère un jeu de données et une classe donnée, la présence d'un certain attribut dans une règle prédisant cette classe peut ne pas surprendre l'utilisateur en raison de l'existence connue de l'utilisateur d'une relation « triviale » liant cet attribut à cette classe. Cependant, ce même utilisateur peut être « surpris » par la présence de cet attribut et celle d'un autre qui, a priori, n'a pas de lien « trivial » avec le premier et qui pourtant caractérisent la classe lorsqu'ils sont envisagés simultanément. Cette dernière notion de « surprise » est reprise plus en détails par le même auteur dans [Frei98b]. Ce dernier adopte une approche objective de la notion de surprise ce qui n'est généralement pas le cas dans la littérature. En effet, ce concept est associé à des considérations le plus souvent subjectives. L'auteur montre d'une part qu'il est possible de formaliser cette notion de façon objective et d'autre part, propose plusieurs idées et méthodes pour définir des mesures objectives de surprise.
Il propose par ailleurs trois critères à optimiser permettant de choisir une mesure :

- La facilité à placer le seuil de la mesure par rapport à sa valeur maximale.

- La linéarité de la mesure.
- La symétrie de la mesure.

Les systèmes qui apprennent à partir d'exemples, expriment souvent le concept appris sous forme d'un ensemble d'exemples caractérisé par une règle.

Etude de l'espace de recherche des règles

N. Radcliffe, dans le contexte de la conception du logiciel GA-MINER [Radc95], s'est intéressé « à ce qui fait un motif intéressant ». En effet, dans un système de fouille de données hautement interactif, l'orientation vers des motifs intéressants est dirigée par l'utilisateur. Cependant, en raison de l'augmentation des données exploitées en fouille de données, il est nécessaire de fournir à ce processus les moyens, c'est-à-dire des mesures objectives, permettant de faire lui-même ces choix afin d'obtenir les motifs les plus intéressants. Ainsi l'auteur identifie certaines caractéristiques qui sont communes aux motifs intéressants :

- En premier lieu, un motif intéressant doit résumer certaines corrélations qui existent dans les données.
- Un motif doit être préferentiellement général bien que l'intérêt et la généralité de celui-ci soient souvent en conflit.
- Le motif doit également ne pas exprimer des tendances qui seraient le fruit de fluctuations non pertinentes dans les données.
- Un motif doit représenter les corrélations non-triviales.
- Un motif doit être approchable c'est-à-dire sous une forme aisément compréhensible par un humain.

Il va sans dire que rassembler toutes ces conditions au sein d'une mesure est une chose complexe et que peu de mesures peuvent s'en prévaloir. Dans ses travaux, l'auteur fournit un certain nombre de fonctions d'évaluation utilisées dans son algorithme GA-MINER.

Il étudie les espaces de recherche de ces fonctions et ce au travers de deux graphes bidimentionnels pour chaque fonction et met en évidence par ce biais des comportements similaires pour certaines d'entre elles. Chaque fonction est représentée en ordonnée. L'axe des abscisses correspondant pour le premier graphe à la cardinalité de A, et pour le second à celle de C. Ces graphes permettent notamment de visualiser des symétries relativement à A et C pour certaines fonctions comme par exemple le χ^2.

Conclusion

Nous avons pu voir l'intérêt d'émettre, comme R. J. Hilderman et al., des principes caractérisant les mesures d'intérêt ; cependant ceux-ci peuvent varier en fonction de l'objectif de l'analyste. P. Lenca propose donc de faire varier les critères de choix, mais une fois les mesures qui nous intéressent sélectionnées, le problème de la manière de les appliquer se pose. Implicitement cela se fait dans une phase de post-traitement sur des règles extraites par un algorithme standard. Enfin, il est intéressant de considérer l'approche selon laquelle plusieurs critères sont satisfaits par une règle et ce au travers de l'optimisation d'une unique mesure. Ces remarques ont motivé notre approche qui rappelons-le est expérimentale. En effet, le comportement des mesures est très nettement influencé par les jeux de données sur lesquels elles sont appliquées, rendant par conséquent une approche analytique inadaptée.

3.2 Etude de l'espace de recherche

Comme nous l'avons évoqué précédemment, nous nous plaçons dans le cadre d'un problème d'optimisation dans lequel nous prenons en compte une ou plusieurs mesures d'intérêt devant être satisfaites par les règles de dépendance extraites. De fait, il est nécessaire d'étudier les espaces de recherche associés à ces mesures. Nous avons utilisé deux méthodes expérimentales d'approximation des espaces de recherche, que nous présentons plus loin. Ces approximations servent à visualiser graphiquement les résultats fournis par l'approche évolutionnaire. Ces résultats sont présentés d'une part dans ce chapitre mais également dans le chapitre 5.
Les espaces de recherche que nous étudions sont composés de règles de dépendance de la forme $A \rightarrow C$ où A et C sont des conjonctions de termes attribut-valeur $Att\ op\ a$ où Att est un attribut, a est une de ses valeurs et $op \in \{<, \leq, =, \geq, >\}$.

3.2.1 Contexte de l'étude

Règles

Nous considérons deux types de règles : les règles *simples* et les règles *généralisées*.

- Les règles simples sont caractérisées par le fait que leur conséquent C est réduit à un seul terme attribut-valeur, qui est le plus souvent de la forme $AttCible = V_C$ avec $AttCible$ un attribut généralement catégoriel, également appelé attribut cible et V_C une valeur du domaine de cet attribut également appelée *classe*. L'opérateur utilisé dans ce terme est généralement l'opérateur $=$, cependant il est envisageable d'employer des opérateurs ensemblistes ou logiques à la condition que le domaine de l'attribut s'y prête. Ces règles sont donc de la forme suivante :

$$\textbf{\textit{SI}}\ A_1\ op_{A_1}\ V_{A_1}\ \textbf{\textit{ET}}\ ...\ \textbf{\textit{ET}}\ A_m\ op_{A_m}\ V_{A_m}\ \textbf{\textit{ALORS}}\ C = V_C$$

Les règles de classification fournies par un algorithme d'induction d'arbre de décision suivent ce modèle syntaxique.

- Les règles généralisées se caractérisent par le fait que leur conséquent est une conjonction d'au moins deux termes attribut-opérateur-valeur. La forme de ces règles est la suivante :

$$\textbf{\textit{SI}}\ A_1\ op_{A_1}\ V_{A_1}\ \textbf{\textit{ET}}\ ...\ \textbf{\textit{ET}}\ A_n\ op_{A_n}\ V_{A_n}$$
$$\textbf{\textit{ALORS}}\ C_1\ op_{C_1}\ V_{C_1}\ \textbf{\textit{ET}}\ ...\ \textbf{\textit{ET}}\ C_p\ op_{C_p}\ V_{C_p}$$

Les règles d'association suivent ce modèle syntaxique.

Jeux de données

Au cours de nos expérimentations nous avons utilisé les jeux de données disponibles sur le site UCI de l'University of California, Irvine [UCI] ainsi qu'un jeu de données fourni par la fondation SFR-Cegetel. Le tableau 3.1 résume les caractéristiques de ces jeux de données. Nous illustrons dans le cadre de ce mémoire nos résultats au travers de trois d'entre eux aux caractéristiques différentes. Les critères ayant guidé nos choix quant à ces jeux de données sont :

- Leur sémantique qui doit être aisément compréhensible, car de celle-ci dépend l'interprétation des règles que nos algorithmes génétiques vont extraire.

- Leur nombre d'instances et d'attributs afin de diversifier les types de jeux de données.

- Leur *Concentration d'erreur* [Weis00b] qui est définie comme étant une mesure permettant d'apprécier à quel point les règles rares associées à un jeu de données sont responsables des erreurs de classement dans un classifieur. Ainsi, nous avons sélectionné deux bases offrant pour l'une une concentration d'erreur élevée, pour l'autre une concentration d'erreur sensiblement plus faible.

Compte tenu de ces critères et de l'ensemble de jeu de données à notre disposition, voici les trois que nous avons sélectionnés :

- Le premier, de petite dimension (si on considère les objectifs de la fouille de données), est le jeu de données « Vote » issu de UCI et dont nous rappelons les caractéristiques principales. Il s'agit d'un jeu de données contenant des descriptions d'hommes politiques aux USA selon 17 attributs binaires, et composé

de 435 instances. Ce jeu de données est connu comme étant corrélé. Son rôle premier est de servir au développement ou/et au test d'outils de classement. Sa sémantique est aisée.

- Le second jeu de données que nous utilisons est le jeu de données « Abonnés » fourni par la fondation SFR-Cegetel. Il contient la description de 14945 clients selon 19 attributs dont 17 attributs de type réel, un attribut binaire qui est celui de la classe et un attribut catégoriel. Ce jeu a été utilisé pour étudier le phénomène d'attrition[5], c'est-à-dire pour déterminer les clients susceptibles de résilier leur contrat auprès de l'opérateur téléphonique. Nous utilisons ce jeu de données en raison de ses dimensions assez grandes relativement aux autres jeux de données « benchmark ». De plus, sa sémantique est facilement compréhensible.

- Enfin, le troisième jeu de données est également un jeu de données de UCI. Il s'agit de « Sonar » dont les caractéristiques sont les suivantes : il contient la description de 208 « motifs » selon 61 attributs dont 60 sont réels et un est binaire. Les attributs réels représentent des niveaux d'énergie pour des bandes de fréquences données. La sémantique de ce jeu est néanmoins difficile, cependant il permet de mettre en évidence des résultats intéressants que nous présentons dans les deux chapitres suivants.

Nous rappelons dans les tableaux 3.2, 3.3 et 3.4, les types des attributs et éventuellement leur sémantique pour les jeux de données « Abonnés », « Sonar » et « Vote », respectivement, sur lesquels nous avons obtenu les résultats présentés dans cette thèse.

Nous avons sélectionné dans le cadre de ce mémoire des mesures dont la signification est aisément compréhensible par l'utilisateur. Les sections suivantes présentent respectivement la densité d'états des mesures, la forme des règles, la variation des valeurs, et enfin les corrélations et antagonismes existant entre certaines mesures.

3.2.2 Approximation des espaces de recherche

Les espaces de recherches que nous considérons étant très vastes, il est extrêmement difficile d'en avoir une « image » fidèle car un parcours exhaustif n'est généralement pas envisageable. Aussi, approximer au mieux ces espaces, afin de connaître les distributions des mesures, permet de se rendre compte qu'il existe de bonnes règles qui sont rares. Cet aspect nécessite la mise en œuvre de méthodes permettant d'obtenir de bonnes approximations.
Pour ce faire, nous utilisons des méthodes non déterministes permettant d'obtenir des approximations de ces espaces. Ces dernières reposent sur la notion de densité d'états dont la définition est la suivante :

[5]L'attrition est dénommé « churn » en anglais.

Nom du jeu de données	*Nb. instances*	*Attr.*	*Attr. bin.*	*Attr. réels*	*Attr. ent.*	*Attr. cat.*
Adult	48842	15	2	0	0	13
Abonnés	**14945**	**19**	**1**	**17**	**0**	**1**
Breast-cancer	286	10	4	0	0	6
Credit-a	692	16	5	6	0	5
Credit-g	1000	21	3	7	0	11
Dermatology	358	35	1	0	1	33
Iris	150	5	0	4	0	1
Labor	57	17	4	8	0	5
Mushroom	8124	23	5	0	0	18
Nursery	12960	9	1	0	0	8
Sonar	**208**	**61**	**1**	**60**	**0**	**0**
Vote	**435**	**17**	**17**	**0**	**0**	**0**
Zoo	101	18	15	0	1	2

TAB. 3.1 – *Caractéristiques des différents jeux de données utilisés. Les jeux de données utilisés dans ce mémoire sont les jeux « Abonnés », « Sonar » et « Vote ».*

Soit un espace de recherche (S, f) où S est l'ensemble des configurations et f la fonction qui associe à chaque configuration s une valeur réelle. Dans notre cas, une configuration est une règle. La *densité d'états* (DOS) mesure la fréquence de chaque valeur de la fonction dans l'espace de recherche. La définition de cette mesure est indépendante de la relation de voisinage et du processus de recherche. Par conséquent, cette mesure est caractéristique de l'espace de recherche, même si l'estimation de DOS comme nous allons le voir ci-dessous peut faire appel à un voisinage ou à un processus de recherche.

Pour les espaces de petite taille, la densité d'états peut être obtenue de façon exacte par énumération exhaustive de l'espace des configurations. Pour les espaces de grande taille, l'énumération complète est irréalisable. Le seul recours est de procéder par approximation. L'estimation peut être réalisée par approximation analytique sur certains problèmes ou encore par approximation expérimentale.

Nous utilisons deux méthodes que nous avons adaptées au cas des règles simples et généralisées. La première qui est la plus naïve, peut ne pas trouver certaines règles lorsque la distribution de celles-ci dans l'espace n'est pas uniforme, il s'agit de la méthode aléatoire. La seconde, plus élaborée, permet de pallier ce défaut. Cette méthode est dénommée méthode de MÉTROPOLIS [Metr53]. Nous avons utilisé la méthode aléatoire dans la suite du mémoire, car compte tenu des jeux de données considérés, les

Pos.	*Attribut*	*Type*	*Sémantique*
1	tarif	discret	Type de tarif
2	âge	continu	Age de l'abonné
3	semRest	continu	Nombre de semaines restantes
4	txwe	continu	% des appels sortants en week-end
5	mois	continu	Ancienneté du contrat en mois
6	jApresChgt	continu	Nombre de jours depuis dernier changement
7	txFixe	continu	% d'appels sortants vers poste fixe
8	npAppsSC	continu	Nombre moyen d'appels vers service client par mois
9	nbAppsE	continu	Nombre moyen d'appels entrants par mois
10	duréeS	continu	Durée moyenne des appels sortants par mois
11	duréeE	continu	Durée moyenne des appels entrants par mois
12	mc	continu	Montant des communications hors forfait
13	sms	continu	Nombre moyen de SMS par mois
14	infoC	continu	Nombre moyen d'appels à Info Conso par mois
15	nbS	continu	Nombre moyen d'appels sortants par mois
16	nbCorr	continu	Nombre de correspondants entrants différents par mois
17	txEchecs	continu	% des échecs des appels sortants par mois
18	resil	discret	Demande de résiliation
19	CP	discret	Code postal

TAB. 3.2 – *Sémantique des attributs du jeu de données « Abonnés ».*

Pos.	*Attribut*	*Type*	*Sémantique*
1 à 60	A_1 à A_{60}	continu	Chacun des 60 attributs représente l'énergie à l'intérieur d'une bande de fréquence particulière intégrée sur une certaine durée dans le temps
61	classe	binaire	Type de terrain

TAB. 3.3 – *Sémantique des attributs du jeu de données « Sonar ».*

deux méthodes fournissent des résultats similaires. La méthode de MÉTROPOLIS est longue à mettre en œuvre et nécessite un paramétrage difficile à déterminer. Ces deux méthodes permettent de générer des règles *simples* ou *généralisées*. Nous présentons ci-dessous la manière dont ces règles sont générées et évaluées par la méthode naïve et par la méthode de MÉTROPOLIS :

- **Génération du conséquent.** Ceci concerne uniquement les règles généralisées. En effet, les règles simples ont toutes le même conséquent de longueur un fixé

Pos.	*Attribut*	*Type*
1	class	binaire
2	handicapped-infants	binaire
3	water-project-cost-sharing	binaire
4	adoption-of-the-budget-resolution	binaire
5	physician-fee-freeze	binaire
6	el-salvador-aid	binaire
7	religious-groups-in-schools	binaire
8	anti-satellite-test-ban	binaire
9	aid-to-nicaraguan-contras	binaire
10	mx-missile	binaire
11	immigration	binaire
12	synfuels-corporation-cutback	binaire
13	education-spending	binaire
14	superfund-right-to-sue	binaire
15	crime	binaire
16	duty-free-exports	binaire
17	export-administration-act-south-africa	binaire

TAB. 3.4 – *Sémantique des attributs du jeu de données « Vote ».*

par l'utilisateur avant la génération aléatoire. Dans un premier temps, un nombre aléatoire non nul de termes définit le conséquent de la règle. Concernant les règles généralisées, ce nombre aléatoire peut au maximum être égal au nombre d'attributs du jeu de données moins un. Pour chaque terme, une valeur appartenant au domaine de l'attribut est ensuite générée de façon aléatoire. Enfin, un opérateur est également généré aléatoirement et ce en adéquation avec la valeur précédemment générée. Si cet attribut est entier ou réel, l'opérateur peut être $<$, $\leq$, $=$, $\geq$ ou $>$. Si la valeur est la borne inférieure ou supérieure du domaine de l'attribut correspondant au terme, l'opérateur ne peut être respectivement que $<$ et $>$. Dans le cas d'un attribut binaire ou catégoriel l'opérateur est $=$.

- **Génération de l'antécédent.** Un nombre aléatoire de termes est généré de la même manière que pour le conséquent des règles généralisées.

- **Evaluation de la règle.** La règle est ensuite évaluée sur le jeu de données et les mesures utilisées pour générer le graphe sont calculées.

Voici les descriptions des deux méthodes d'approximation des espaces de recherche.

Méthode du tirage aléatoire

Cette méthode consiste à approcher l'espace de recherche selon l'algorithme présenté à la figure 3.9. Il s'agit de générer de façon aléatoire des règles.

```
1)  Algorithme TirageAléatoire
2)  Entrée :
3)      entier N : nombre de règles à générer
4)  Sortie :
5)      ensemble E : approximation de l'espace de recherche
6)  Données :
7)      ensemble R : ensemble des règles
8)      règle r : une règle générée
9)      entier i : compteur
10) Début
11)     E ← ∅
12)     i ← 0
13)     Tant que (i < N) faire
14)         i ← i + 1
15)         r ← GénérerUneRègleAléatoirement(R)
16)         E ← E ∪ {r}
17)     Fin tant que
18)     Retourner E
19) Fin
```

FIG. 3.9 – *L'algorithme du tirage aléatoire.*

Méthode de MÉTROPOLIS

La méthode de MÉTROPOLIS [Metr53] est à la base de la méthode des températures. Cette dernière permet d'approcher entre autre, une des mesures liées aux espaces de recherche qui est la densité d'état dans le but de caractériser ces espaces. Cette méthode se base sur une analogie thermodynamique. La méthode de MÉTROPOLIS appliquée à un jeu de données, fonctionne de la façon suivante : le processus est initialisé avec une règle r_1 aléatoire évaluée sur le jeu de données. La règle suivante r_2 déterminée également de façon aléatoire est acceptée avec une probabilité p définie par :

$$p = \begin{cases} 1 \ si \ \delta_i \geq 0 \\ e^{\frac{\delta_i}{T}} \ si \ \delta_i < 0 \end{cases} \tag{3.11}$$

où $\delta_i = mesure_i(r_2) - mesure_i(r_1)$ et $mesure_i(r)$ est la $i^{ème}$ mesure prise en compte. Nous nous plaçons cependant pour l'instant dans le cadre de la prise en compte d'une seule mesure. Le paramètre T est appelée la température et est déterminé de façon empirique. L'algorithme est appliqué pour chaque mesure et les ensembles de points obtenus sont superposés.
Le choix de la prochaine règle r_2 peut être réalisé de différentes manières : soit l'obtention de cette règle est indépendante de la règle précédente r_1, soit elle est prise au hasard parmi l'ensemble des règles voisines de r_1 définies par une relation de voisinage. Pour ne pas biaiser l'approximation de l'espace de recherche (espace des règles), il est possible d'éviter d'utiliser une relation de voisinage. Cependant ceci ralentit considérablement le traitement, car générer une nouvelle règle r_2 devient alors une opération coûteuse. La figure 3.10 présente l'algorithme de MÉTROPOLIS.

Le réglage du paramètre T, qui est le paramètre permettant d'échantillonner l'espace de recherche, doit s'effectuer de manière empirique.
Le choix d'une relation de voisinage biaise l'approximation de l'espace de recherche. Cependant l'absence de relation de voisinage engendre un temps de calcul prohibitif. Le compromis consiste à choisir une relation de voisinage telle que celle-ci n'influence pas les résultats de la méthode de MÉTROPOLIS. Le premier graphe de la figure 3.11 correspond au résultat d'exécution de la méthode aléatoire sur le jeu de données « Vote » pour 5000 règles générées. Le second graphe correspond au résultat d'exécution de l'algorithme de MÉTROPOLIS pour le même nombre de règles avec une température égale à 0.075. Cette température comme nous l'avons dit est déterminée de façon empirique et les résultats obtenus pour des températures plus faibles ou plus élevées mettent en évidence des résultats moins bons. En effet, nous recherchons à approximer l'espace de recherche et notamment les « zones » contenant des points se situant aux alentours de ce que l'on nomme la frontière de Pareto, concept que nous définissons au chapitre 5. Cette frontière correspond dans les graphes (a) et (b) aux points situés à la frontière du nuage dans le quadran du haut à droite. On distingue sur ces graphes que cet ensemble est mieux approximé sur le graphe (b) produit par l'algorithme de MÉTROPOLIS que par celui généré par la méthode aléatoire, c'est-à-dire que la densité des points est plus importante sur le graphe (b).

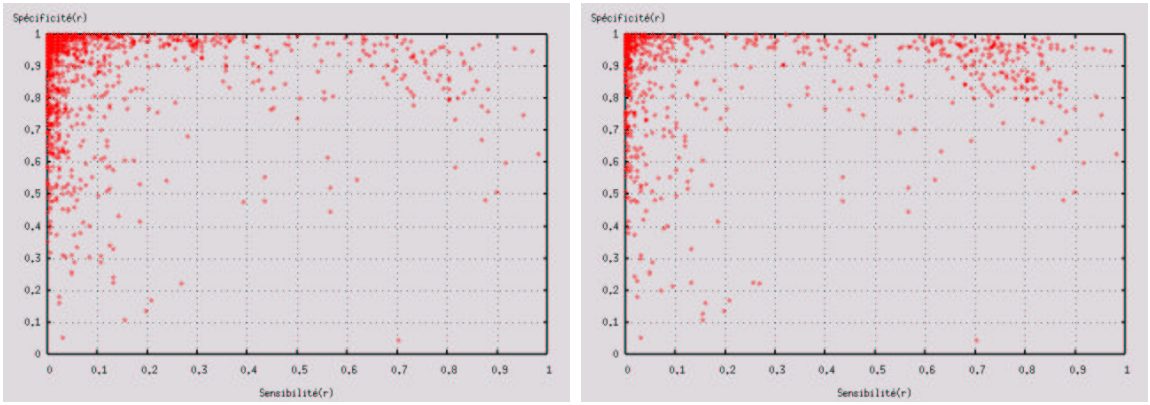
3.2.3 Densité d'états

L'application des méthodes d'approximation des espaces de recherche fournissent des nuages de points dont nous nous servons pour obtenir diverses informations concernant les règles de ces espaces. Nous illustrons ici un de ces aspects au travers de la densité d'états de ces espaces qui consiste à visualiser sur un graphe à deux dimensisons, le nombre de règles (de points) ayant une valeur donnée. Par exemple le graphe de la figure 3.12 qui correspond au nombre de règles simples ayant une valeur donnée de la *Sensibilité* sur le jeu de données « Vote », présente une particularité que l'on retrouve sur plusieurs mesures et sur plusieurs jeux de données. On peut remarquer dans ce graphe une très nette diminution du nombre de règles pour des valeurs élevées

```
1)  Algorithme MÉTROPOLIS
2)  Entrée :
3)      entier nbRègles : nombre de règles à générer
4)  Sortie :
5)      ensemble E : approximation de l'espace de recherche
6)  Données :
7)      entier nbMesures : nombre de mesures considérées
8)      ensemble R : ensemble des règles
9)      v : relation de voisinage
10)     règles r1 et r2 : les règles générées
11)     entiers i et j : compteurs
12)     réel δ, T, aléat : distance, température et réel aléatoire
13) Début
14)     i ← 0
15)     Tant que (i < nbMesures) faire
16)         r1 ← GénérerUneRègleAléatoirement(R)
17)         E ← {r1}
18)         j ← 0
19)         Tant que (j < nbRègles) faire
20)             //Choix d'une règle parmi les voisins v(r1)
21)             //ou de façon indépendante de r1 :
22)             r2 ← GénérerUneRègle(R, v)
23)             δ ← mesure_i(r2) − mesure_i(r1)
24)             aléat ← GénérerUnRéelAléatoirement(O, 1)
25)             Si (δ > 0) alors
26)                 r1 ← r2
27)                 E ← E ∪ {r1} ; j ← j + 1
28)             Sinon si (aléat < e^(δ/T)) alors
29)                 r1 ← r2
30)                 E ← E ∪ {r1} ; j ← j + 1
31)             Fin si
32)         Fin tant que
33)         i ← i + 1
34)     Fin tant que
35)     Retourner E
36) Fin
```

FIG. 3.10 – *L'algorithme de* MÉTROPOLIS.

de la mesure. Celles-ci peuvent être intéressantes. Sur le jeu de données « Abonnés »,



(a) (b)

FIG. 3.11 – *Comparatif entre les résultats de la méthode aléatoire (a) et de la méthode de* MÉTROPOLIS *(b) sur le jeu de données « Vote » pour 5000 règles générées. On distingue notamment en haut à droite du graphe (b), une densité plus importante de points que dans le graphe (a).*

on constate les mêmes caractéristiques avec le coefficient de *Sebag* et la *Conviction*. La figure 3.13 illustre cela.

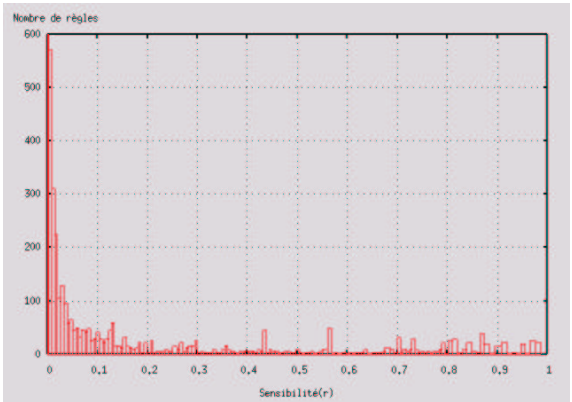
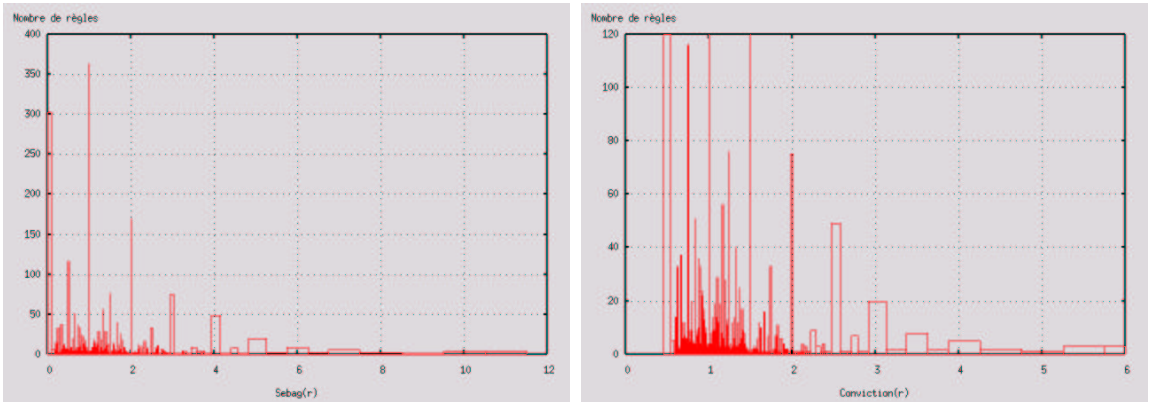


FIG. 3.12 – *Densité des règles simples basées sur la Sensibilité sur le jeu de données « Vote ».*

Le graphe de la figure 3.14 est un contre-exemple. Nous avons dans ce graphe des règles correspondant à une valeur élevée de la mesure et présentes en grand nombre.



(a) (b)

FIG. 3.13 – *Densité des règles simples basées sur le coefficient de Sebag (a) et sur la Conviction (b) sur le jeu de données « Abonnés ».*

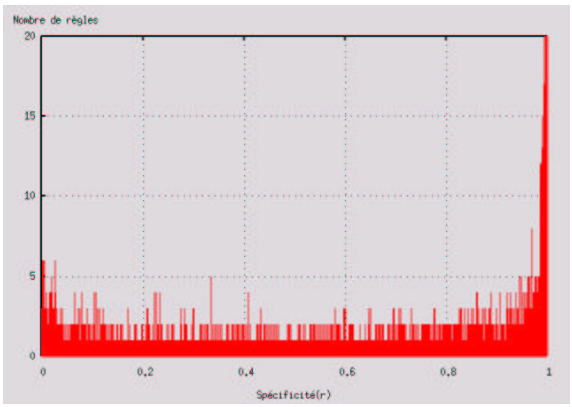


FIG. 3.14 – *Densité des règles simples basées sur la Spécificité sur le jeu de données « Abonnés ».*

3.2.4 Forme des règles

Nous illustrons ici une autre caractéristique liée aux règles générées par les méthodes d'approximation des espaces de recherche. Il s'agit de « la forme des règles » et qui consiste à représenter sur un graphe en deux dimensions la longueur moyenne des règles ayant une valeur d'une mesure donnée. Les graphes des figures 3.15, 3.16, 3.17 et 3.18 présentent cette étude.

Le graphe de la figure 3.15, obtenu sur le jeu de données « Vote » représente la longueur moyenne des antécédents des règles simples basées sur le $Lift$. On peut

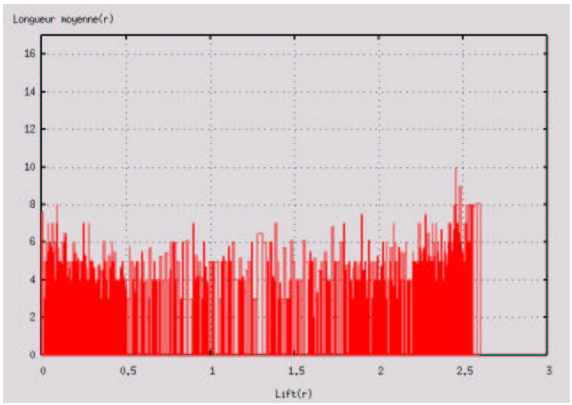
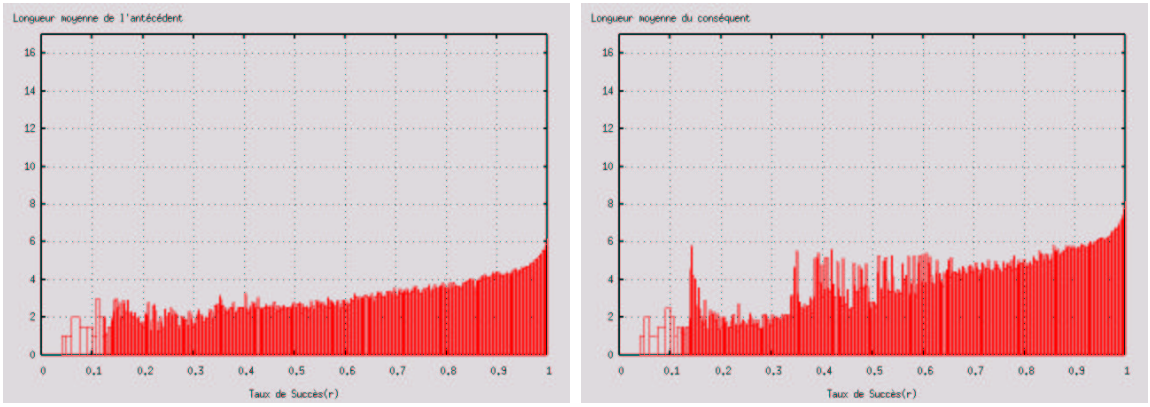


FIG. 3.15 – *Longueur des antécédents des règles simples basées sur le $Lift$ et obtenues à partir du jeu de données « Vote ».*

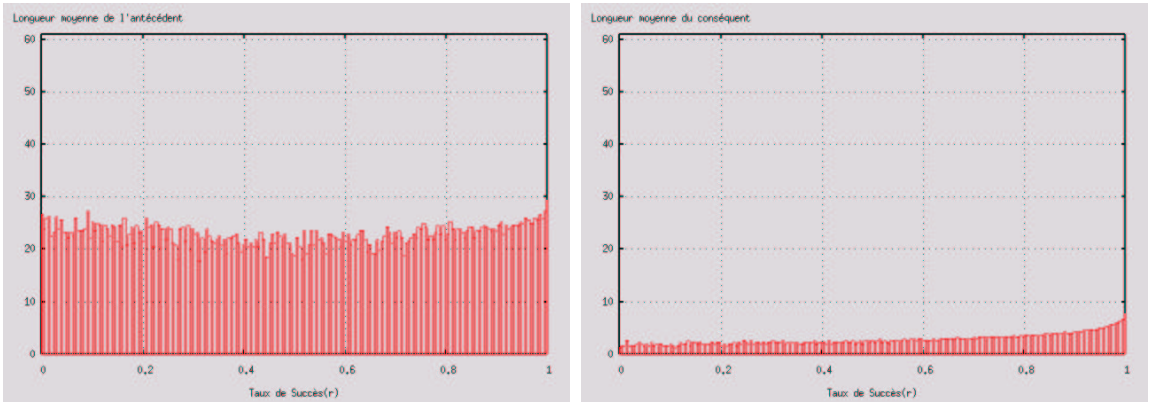
observer dans ce graphe, qu'il existe des règles ayant une valeur de $Lift$ relativement élevée et ayant un antécédent assez long.



(a) (b)

FIG. 3.16 – *Longueur des antécédents (a) et des conséquents (b) des règles généralisées basées sur le $Taux\ de\ succès$ et obtenues sur le jeu de données « Vote ».*

Les graphes des figures 3.16 et 3.17 relatifs au *Taux de succès* respectivement sur les jeux de données « Vote » et « Sonar », présentent chacun la caractéristique suivante : globalement, il y a une corrélation entre la mesure et la longueur ; plus une règle est longue, plus elle est performante. Il existe des règles ayant des longueurs de conséquent (mais également d'antécédent) relativement élevées pour des valeurs



(a) (b)

FIG. 3.17 – *Longueur des antécédents (a) et des conséquents (b) des règles généralisées basées sur le Taux de succès et obtenues sur le jeu de données « Sonar ».*

du *Taux de succès* également élevées. Cependant, le graphe (b) de la figure 3.17 se distingue des autres par le fait qu'il est relativement plat. On peut noter que les algorithmes d'induction d'arbre de décision dérivés de C4.5 ne trouvent pas ce genre de règles en raison de l'élagage qu'ils effectuent. Le graphe de la figure 3.18 montre par ailleurs un contre-exemple dans le sens où il existe des règles courtes dont la *Sensibilité* est élevée.

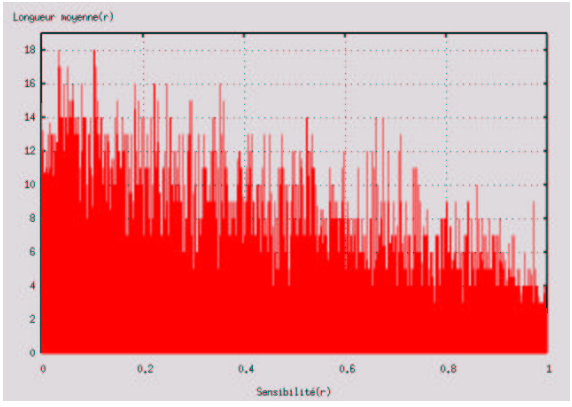


FIG. 3.18 – *Longueur des antécédents des règles simples basées sur la Sensibilité et obtenues à partir du jeu de données « Abonnés ».*

3.2.5 Variation des valeurs

Afin de mettre en évidence des maxima locaux dans les espaces de recherche des mesures, nous utilisons l'approche suivante qui consiste à représenter pour une mesure donnée, les valeurs de cette mesure en ordonnée pour un nombre de règles générées aléatoirement, et en abscisse la distance de ces règles par rapport à une règle dite de référence qui est la meilleure règle trouvée par l'algorithme d'induction d'arbres de décision. Les graphes des figures 3.19, 3.20 et 3.21 présentent cette seconde approche respectivement sur les jeux de données « Vote », « Abonnés » et « Sonar ».
Sur le graphe de la figure 3.19 relatif au coefficient de *Sebag* sur le jeu de données « Vote », on peut constater qu'il existe de nombreuses règles, ayant un coefficient de *Sebag* plus élevé que la règle de référence, et éloignées de cette règle selon la distance de Hamming. Ces règles ne sont pas trouvées par l'algorithme d'induction d'arbre de décision en raison de l'heuristique utilisée par ce dernier. Nous faisons figurer sur chacun de ces graphes l'ensemble des règles ayant en commun l'antécédent de la règle de référence. On distingue sur ce premier graphe que les règles non élaguées sont plus mauvaises que la règle de référence. Cependant, cette dernière a une valeur beaucoup plus basse que certaines règles correspondants aux maxima locaux.

Voici comment nous définissons la distance d_H de deux règles simples r et r' dans un premier temps. Pour chaque attribut du jeu de données, si celui-ci est présent dans r et absent de r' (et vice et versa), alors d_H est incrémentée. Notons que dans le cas des règles généralisées, une condition basée sur un attribut peut être présente dans l'antécédent de r et également dans le conséquent de r' (étant donné que ces règles ont un conséquent qui n'est pas limité à un attribut comme celui des règles simples). Dans ce cas, la distance d_H peut être incrémentée si l'attribut de la condition de chaque règle appartient pour l'un à l'antécédent et pour l'autre au conséquent.

Le même phénomène sur la $Confiance$ et le $Taux\ de\ succès$ sur le jeu de données « Abonnés » est visible dans les deux graphes de la figure 3.20. Ici aussi, des règles lointaines sont sensiblement meilleures que la règle de référence.

Enfin, sur le jeu de données « Sonar » et relativement à la *Conviction* (figure 3.21), on distingue des règles nettement meilleures que la règle de référence. Notons que dans les quatre exemples présentés précédemment, nous avons vérifié que les règles meilleures et éloignées de la règle de référence ne sont pas trouvées par l'algorithme d'induction d'arbres de décision.
Ainsi, les algorithmes standards de par leur architecture sont dans l'incapacité de trouver certaines règles « intéressantes » lorsque l'on envisage une seule mesure. La section suivante explore l'approche basée sur deux mesures.

3.2.6 Corrélations et antagonismes

Il peut être utile dans le cadre de l'extraction de règles de dépendance, de mesurer ces motifs selon une ou plusieurs mesures de qualité. L'objectif de cette section est

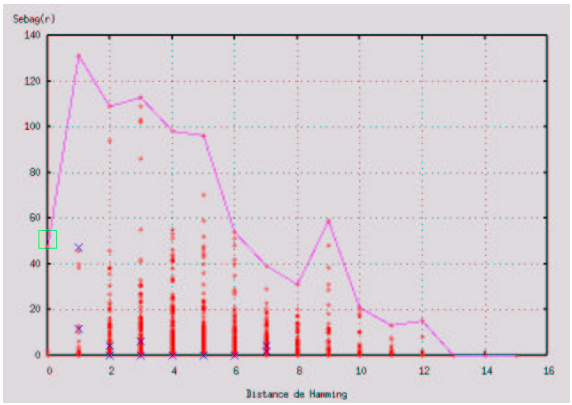
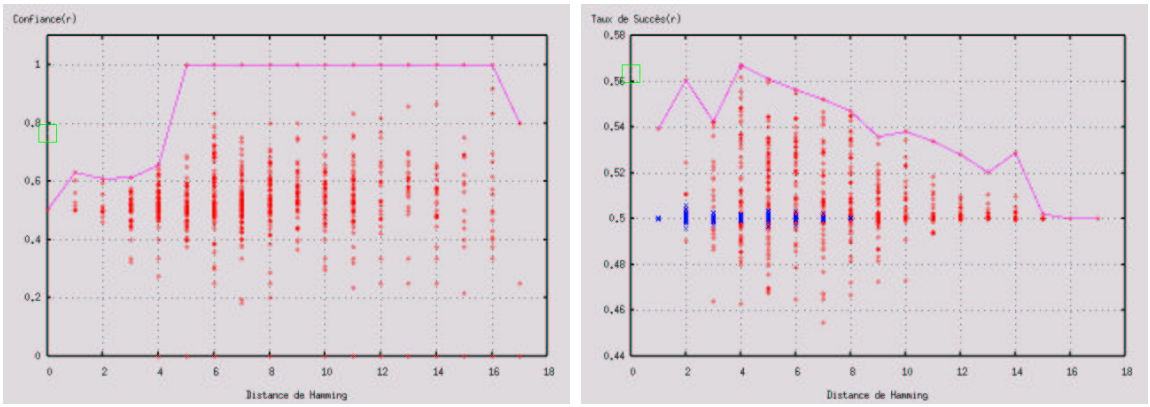


FIG. 3.19 – *Distance de Hamming par rapport à la règle de référence générée par l'algorithme d'induction d'arbre de décision et un ensemble de règles aléatoires basées sur le coefficient de Sebag sur le jeu de données « Vote ». Le carré correspond à la règle de référence et les croix aux règles de l'arbre non élagué.*



(a) (b)

FIG. 3.20 – *Distances de Hamming par rapport à deux règles de référence générées par l'algorithme d'induction d'arbre de décision et un ensemble de règles aléatoires basées sur la Confiance (a) et le Taux de succès (b) sur le jeu de données « Abonnés ». Le carré correspond à la règle de référence et les croix aux règles de l'arbre non élagué.*

d'étudier cet aspect. Nous présentons ici les nuages aléatoires de certains n-uplets de mesures. Ces nuages vont permettrent d'étudier et de valider de façon expérimentale les résultats fournis par les algorithmes génétiques que nous mettons en œuvre dans les deux chapitres suivants. Comme nous l'avons évoqué précédemment, il est souvent

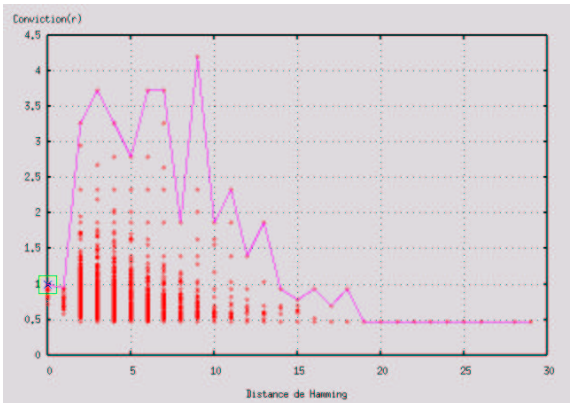


FIG. 3.21 – *Distance de Hamming par rapport à une règle de référence générée par l'algorithme d'induction d'arbre de décision et un ensemble de règles aléatoires basées sur la Conviction sur le jeu de données « Sonar ». Le carré correspond à la règle de référence et les croix aux règles de l'arbre non élagué.*

nécessaire de prendre en compte plusieurs mesures. Nous présentons ici, sur la donnée des mesures énoncées, plusieurs couples sur lesquels nous illustrons nos expérimentations dans ce mémoire.

Chaque figure représente, pour un couple de mesures données, trois graphes. Celui de gauche est relatif au jeu de données « Vote », celui de droite au jeu de données « Abonnés » et celui du bas au jeu de données « Sonar ». Nous pouvons aisément constater les grandes variations de représentations graphiques pour un même couple de mesures suivant le jeu de données en présence. D'une manière générale, un couple de mesures présentant une corrélation particulière sur un jeu de données, ne la présente pas nécessairement sur les autres ce qui justifie cette approche expérimentale. De nombreux couples ne sont pas antagonistes. Citons par exemple le couple $(Sebag, Confiance)$ représenté sur le graphe de la figure 3.22 à partir du jeu de données « Abonnés ». Nous envisageons pour chaque couple de mesures, le cas des règles simples et celui des règles généralisées.

Conviction versus Support

Dans la recherche de règles rares, il est intéressant de prendre en compte l'implication logique des règles par le biais de la *Conviction* ainsi que leur *Support*. Bien entendu, il s'agit de ne garder, parmi les compromis de règles trouvés, que ceux dont le *Support* est inférieur à un seuil fixé par l'utilisateur. De la même manière, il est possible suite à un filtrage sur le *Support*, de ne garder que les règles générales.
A partir des définitions 3.10 et 3.3 de ces mesures, on déduit que celles-ci sont liées par la relation suivante :

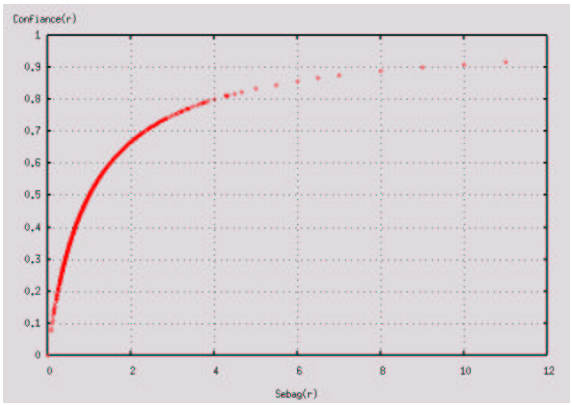


FIG. 3.22 – *Nuage aléatoire composé de règles simples selon les mesures Sebag et Confiance sur le jeu de données « Abonnés ».*

$$\boxed{Support(r) = -\frac{p(\neg C) \times p(A)}{Conviction(r)} - p(A)} \tag{3.12}$$

Le graphe (a) relatif au jeu de données « Vote » pour les règles simples, de la figure 3.23 présente un ensemble « d'alignements » des points. De même sur le graphe (a) du même jeu de données pour les règles généralisées de la figure 3.24. On distingue sur ces figures, que l'allure des graphes dépend fortement des caractéristiques des jeux de données. Ainsi, les graphes associés aux jeux de données « Abonnés » et « Sonar » sont beaucoup plus denses.

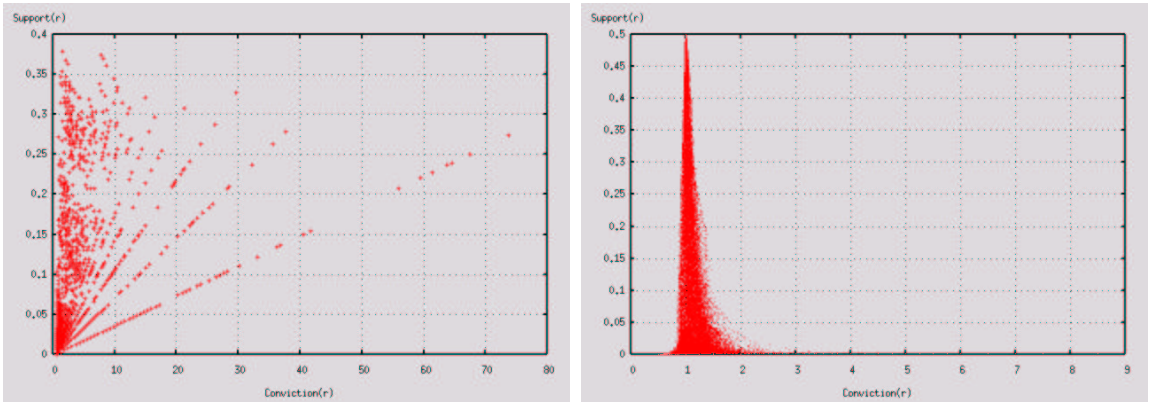
Lift versus Support

De la même manière que dans le cas précédent, il peut être intéressant d'associer la prise en compte du *Support* des règles avec une autre mesure. Le $Lift$ sera d'autant plus grand que le nombre d'instances couvert par le conséquent C des règles sera faible. Cela permet de traiter le second cas relatif aux règles rares, que nous avons mis en évidence au chapitre 3.

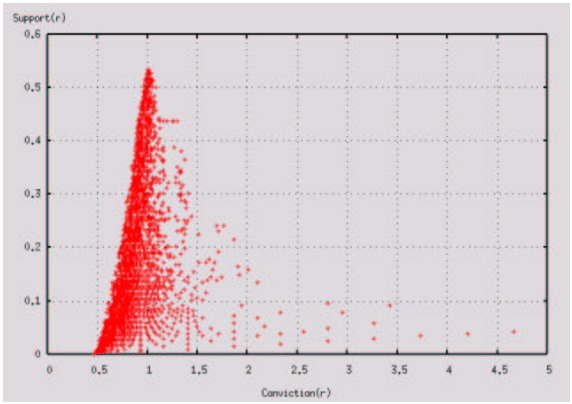
A partir des définitions 3.5 et 3.3 de ces mesures, on déduit la relation suivante entre celles-ci :

$$\boxed{Support(r) = p(A) \times p(C) \times Lift(r)} \tag{3.13}$$

Les graphes de la figure 3.25 concernant les règles simples présentent les représentations graphiques pour ce couple. Concernant les règles généralisées, ce couple ne présente pas d'antagonismes.



(a) (b)



(c)

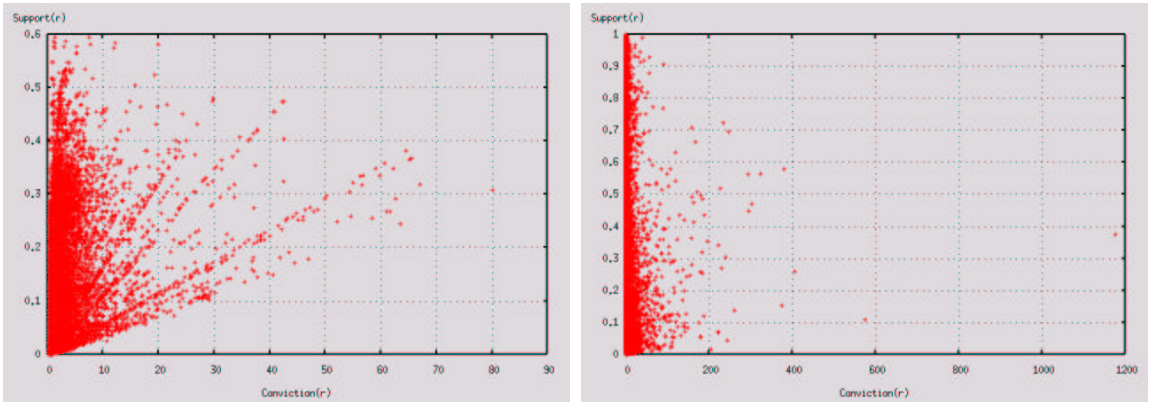
FIG. 3.23 – *Nuages aléatoires composés de règles simples selon les mesures Conviction et Support sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*

Sebag versus RI

Ces deux mesures sont complémentaires dans le sens où le coefficient de *Sebag* prend en compte le nombre de contre-exemples, ce que la mesure RI ne fait pas. Cependant, cette dernière permet de prendre en compte le nombre d'exemples couverts par l'antécédent A des règles ce qui permet la mise en évidence de règles générales.

La relation qui lie ces mesures est la suivante, obtenue à partir des définitions 3.9 et 3.6 :

$$\boxed{RI(r) = |A \cap C| \times Sebag(r) - n \times p(A) \times p(C)} \tag{3.14}$$



(a) (b)

FIG. 3.24 – *Nuages aléatoires composés de règles généralisées selon les mesures Conviction et Support sur les jeux de données « Vote » (a) « Abonnés » (b).*

Cette expression explique ici aussi la présence de points alignés selon des droites dans le graphe (a) relatif au jeu de données « Vote » pour les règles simples de la figure 3.26 ainsi que dans le graphe du même jeu de données pour les règles généralisées de la figure 3.27.

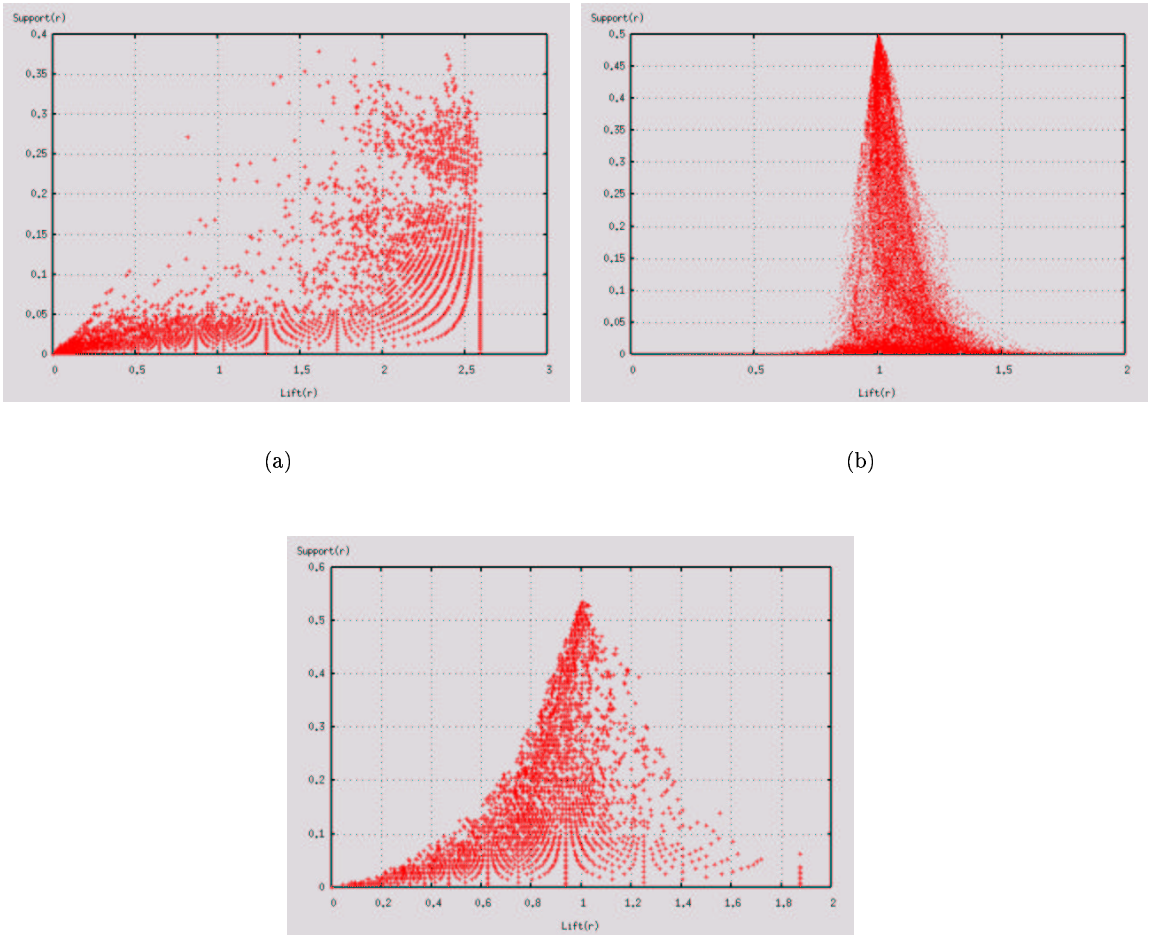
Sebag versus Support

Nous avons utilisé le *Support* pour la recherche de règles rares à la fois avec la *Conviction* pour la prise en compte de l'implication logique et le $Lift$ pour la prise en compte du conséquent des règles. L'utilisation du coefficient de *Sebag* avec cette mesure permet de trouver des règles rares en tenant compte du nombre de contre-exemples.

A partir des définitions 3.9 et 3.3, on déduit que ces mesures sont liées par la relation suivante :

$$\boxed{Support(r) = p(A \cap \neg C) \times Sebag(r)} \tag{3.15}$$

Cette expression explique la présence de points alignés selon des droites passant par l'origine dans le graphe (a) relatif au jeu de données « Vote » pour les règles simples de la figure 3.28 ainsi que dans le graphe relatif au même jeu de données de la figure 3.29.



(a) (b)

(c)

FIG. 3.25 – *Nuages aléatoires composés de règles simples selon les mesures Lift et Support sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*

Sensibilité versus Lift

On déduit la relation suivante entre ces deux mesures à partir des définitions 3.1 et 3.5 :

$$\boxed{Lift(r) = \frac{Sensibilité(r)}{p(A)}} \tag{3.16}$$

Cette expression explique ici aussi la présence de points alignés selon des droites passant par l'origine dans le graphe de gauche de la figure 3.30 concernant les règles simples. Concernant les règles généralisées, aucun jeu de données ne présente d'antagonisme

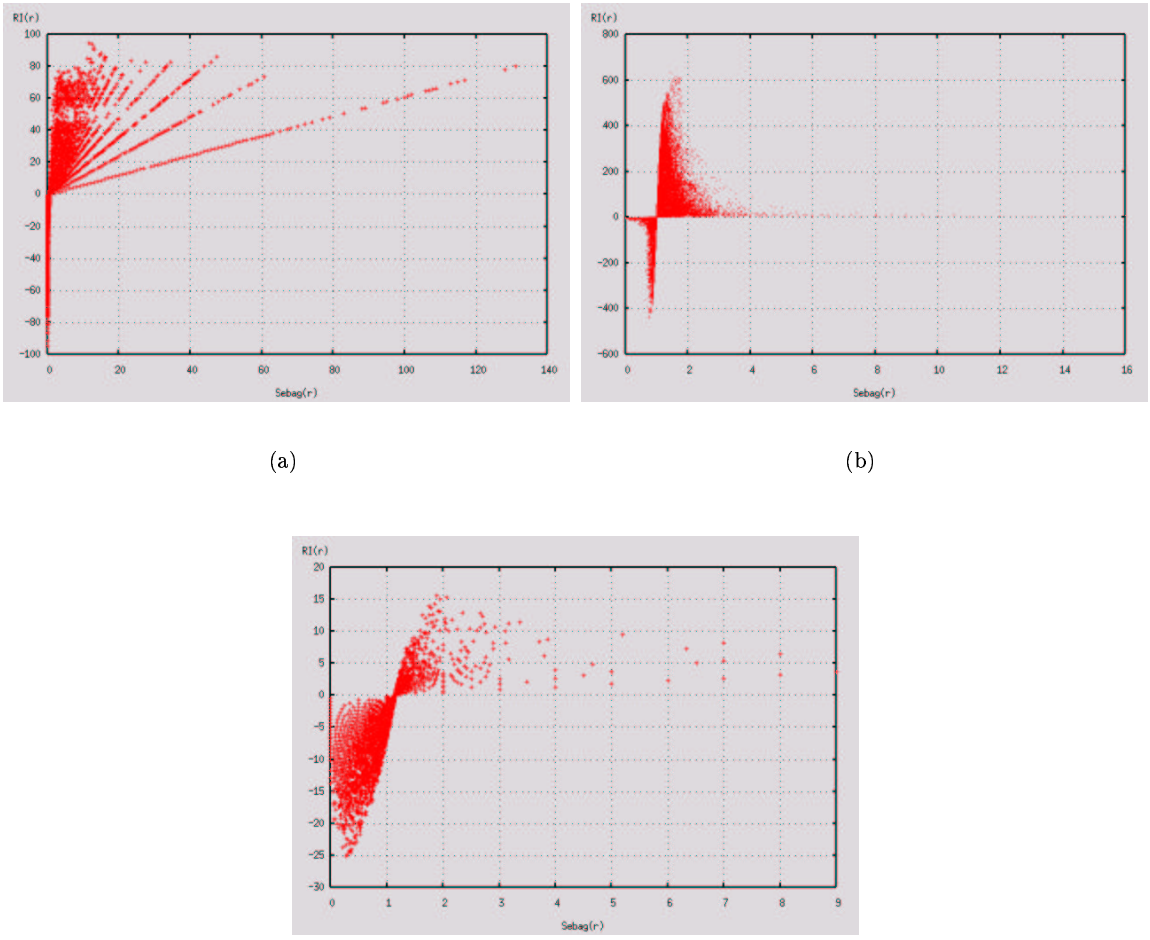


FIG. 3.26 – *Nuages aléatoires composés de règles simples selon les mesures Sebag et RI sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*

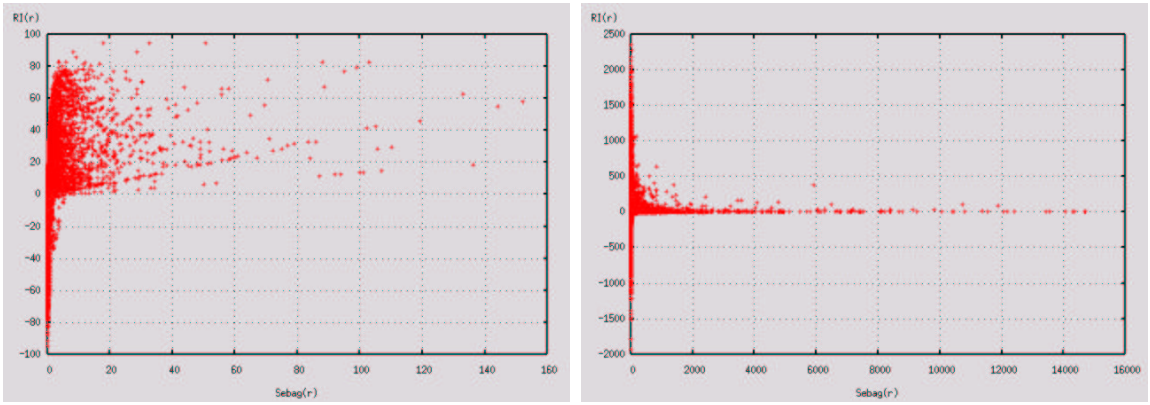
avec ces deux mesures.

Sensibilité versus Sebag

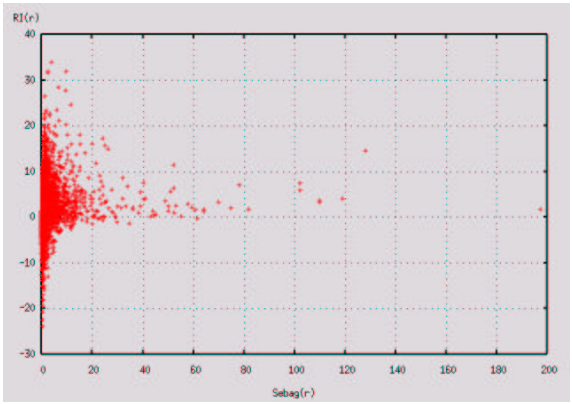
A partir des expressions 3.1 et 3.9 de la *Sensibilité* et du coefficient de *Sebag*, on déduit la relation suivante entre ces deux mesures :

$$\boxed{Sebag(r) = \frac{p(C)}{p(A \cap \neg C)} \times Sensibilité(r)} \tag{3.17}$$

Les graphes de la figure 3.31 et 3.32, respectivement pour les règles simples et généra-



(a) (b)



(c)

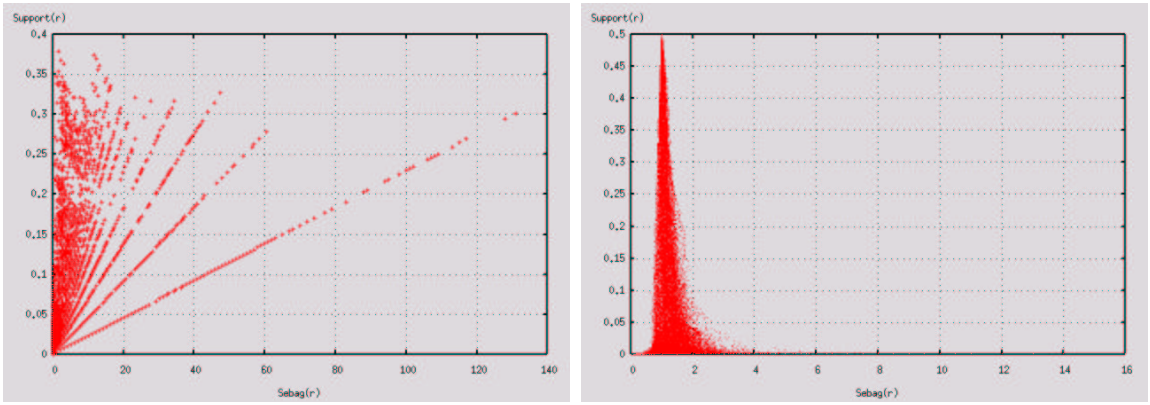
FIG. 3.27 – *Nuages aléatoires composés de règles généralisées selon les mesures Sebag et RI sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*

lisées, présentent les représentations graphiques pour ce couple.

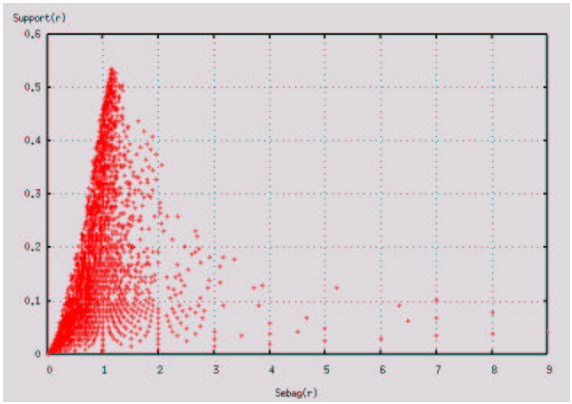
Sensibilité versus Spécificité

L'optimisation simultanée de ces deux mesures, permet de trouver des règles dont l'antécédent A et le conséquent C tendent à se superposer.

Le graphe (a) de la figure 3.33 présente une corrélation négative beaucoup moins marquée que le graphe (b) de cette même figure. D'autre part, les densités des espaces de recherches des couples de mesures pour le jeu de données « Abonnés » sont beaucoup plus denses en raison du fait que les attributs de ce jeu sont pour la plupart réels et



(a) (b)



(c)

FIG. 3.28 – *Nuages aléatoires composés de règles simples selon les mesures Sebag et Support sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*

ceux de « Vote » catégoriels, ce qui augmente considérablement le nombre possible de règles.

3.3 Conclusion

Nous avons pu voir l'intérêt de formuler des principes caractérisant les mesures d'intérêt ; cependant ceux-ci peuvent varier en fonction de l'objectif de l'analyste. Diverses approches ont été adoptées dans la littérature comme celle consistant à faire varier les critères de choix, ce qui n'apporte pas de solutions concernant leur application. Il est intéressant de considérer l'approche selon laquelle plusieurs critères sont satisfaits par

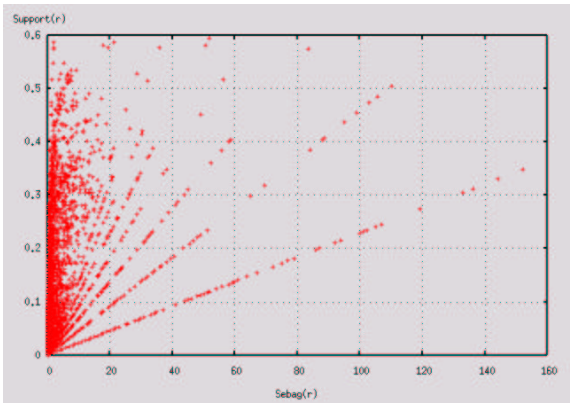
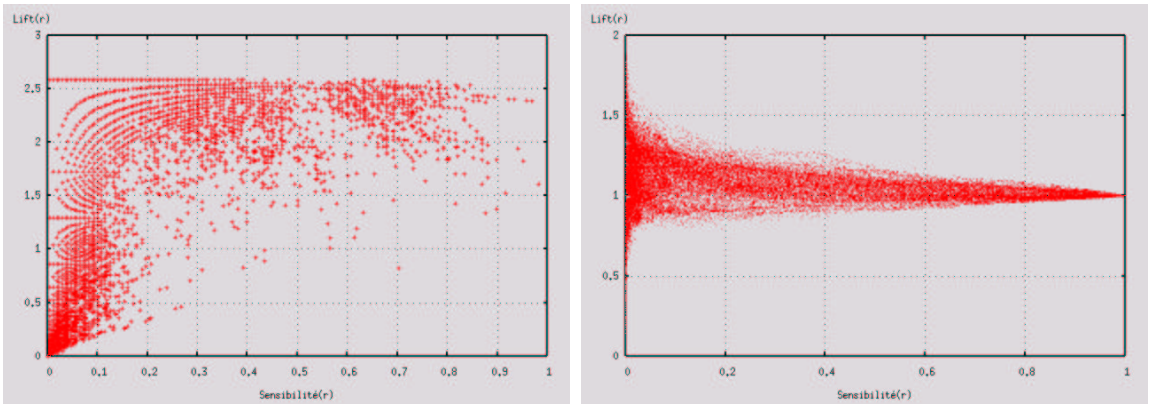
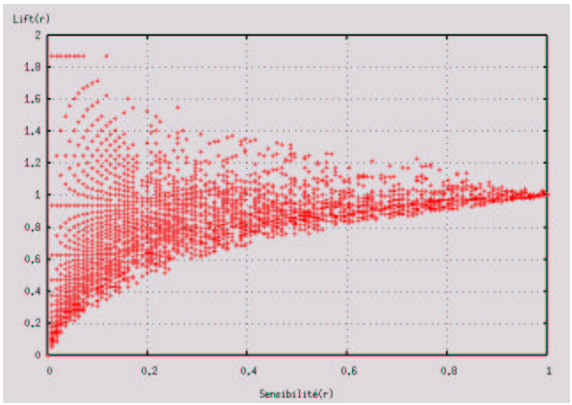


FIG. 3.29 – *Nuage aléatoire composé de règles généralisées selon les mesures Sebag et Support sur le jeu de données « Vote ».*

une règle et ce au travers de l'optimisation d'une unique mesure. Ces remarques ont motivé notre approche qui rappelons-le est expérimentale. En effet, le comportement des mesures est très nettement influencé par les jeux de données sur lesquels elles sont appliquées, rendant par conséquent une approche analytique inadaptée.
Nous envisageons la fouille de données d'un point de vue optimisation ce qui nécessite l'étude des espaces de recherche des mesures à optimiser. Nous avons vu dans ce chapitre diverses approches permettant de mettre en lumière des particularités liées aux mesures sur certains jeux de données, et qui justifient l'approche expérimentale que nous présentons. Ces particularités sont relatives à l'existence d'un nombre restreint de règles ayant une valeur élevée pour une mesure donnée, de même qu'il existe des règles assez longues et « bonnes » relativement à une mesure. Nous avons également mis en évidence par la variation de la syntaxe des règles et de la distance de Hamming (par rapport à une règle de l'algorithme d'induction d'arbres de décision) l'existence de maxima locaux. Enfin, nous avons montré de façon expérimentale l'existence d'antagonismes entre certaines mesures. Le chapitre suivant présente la métaheuristique des algorithmes évolutionnaires permettant d'apporter des solutions à ces divers problèmes.

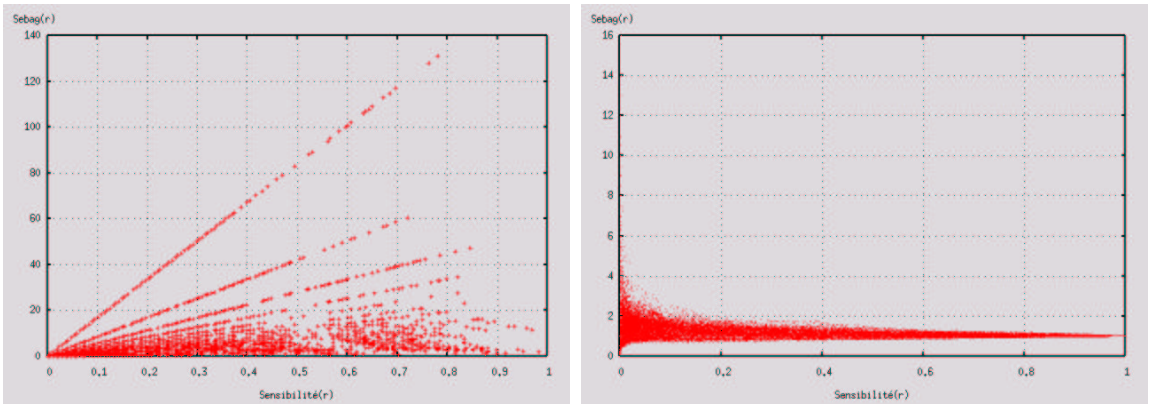


(a) (b)

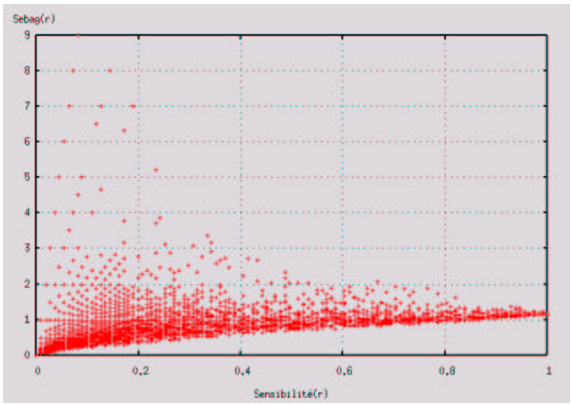


(c)

FIG. 3.30 – *Nuages aléatoires composées de règles simples selon les mesures Sensibilité et Lift sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*



(a) (b)



(c)

FIG. 3.31 – *Nuages aléatoires composés de règles simples selon les mesures Sensibilité et Sebag sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*

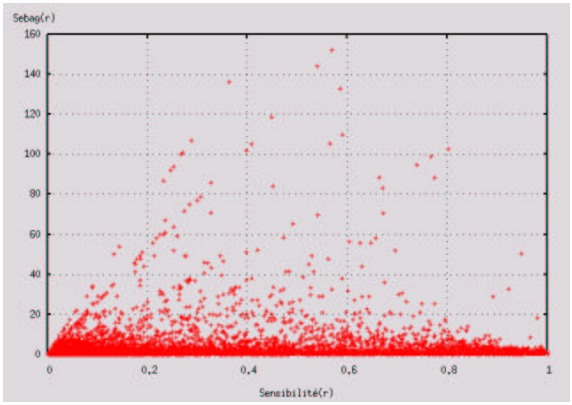
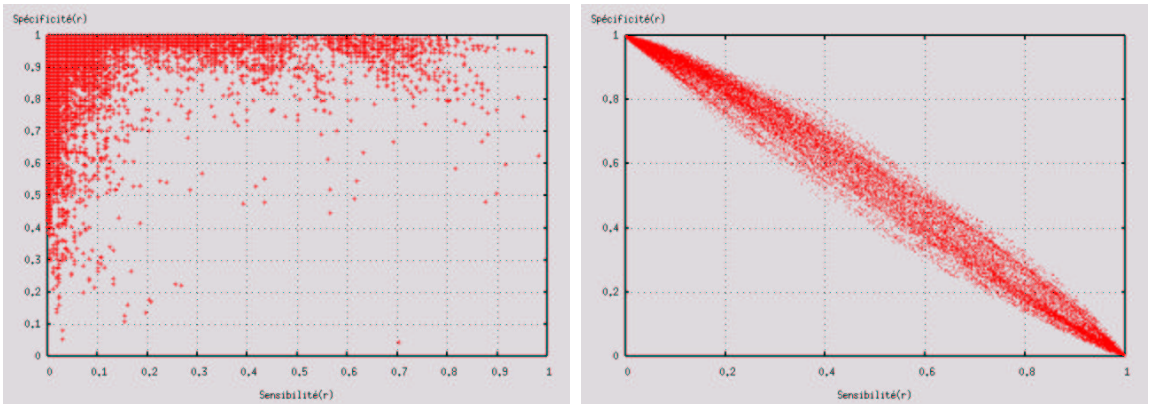
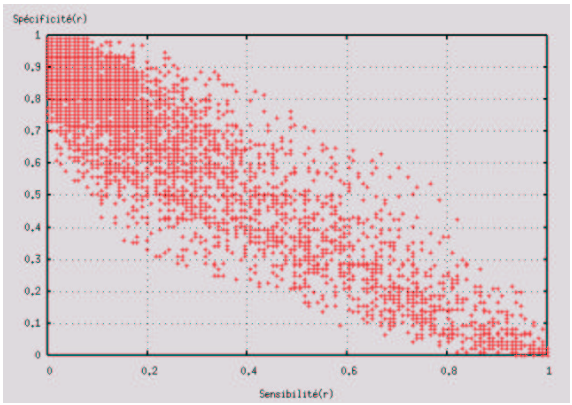


FIG. 3.32 – *Nuage aléatoire composé de règles généralisées selon les mesures Sensibilité et Sebag sur le jeu de données « Vote ».*



(a) (b)



(c)

FIG. 3.33 – *Nuages aléatoires composés de règles simples selon les mesures Sensibilité et Spécificité sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c).*

Chapitre 4

Algorithmes évolutionnaires : méthodes avancées pour la fouille de données

Sommaire

Le chapitre précédent a mis en lumière les particularités des espaces de recherche de règles suivant les jeux de données employés et la nécessité d'avoir recours à une approche d'optimisation pour l'extraction de règles de dépendance. Dans cette optique, le présent chapitre est relatif à la présentation de la métaheuristique que nous

considérons : celle des algorithmes évolutionnaires et en particulier celle des algorithmes génétiques. Nous présentons un état de l'art des travaux existants mettant en œuvre des méthodes évolutionnaires en fouille de données. Ensuite, nous présentons, en nous basant sur les résultats présentés dans le chapitre précédent, les résultats obtenus à l'aide d'une version de l'algorithme génétique que nous avons développé et prenant en compte une seule mesure de qualité pour extraire les règles de dépendance. Le cas de la prise en compte de plusieurs mesures fait l'objet du chapitre suivant.

4.1 Algorithmes évolutionnaires

Les algorithmes évolutionnaires sont des métaheuristiques comprenant trois types de méthodes : les algorithmes génétiques, la programmation génétique et les stratégies d'évolution. La section suivante présente les deux premières catégories d'algorithmes auxquelles nous nous intéressons dans ce mémoire. Les différentes entités constituant les algorithmes génétiques sont abordées avant de conclure sur ce qui fait la force de cette métaheuristique.

4.1.1 Algorithmes génétiques

Les algorithmes génétiques [Holl75], qui font partie des méthodes énumératives probabilistes, sont des algorithmes à l'origine conçus pour traiter des problèmes d'optimisation. Ils ont été introduits et développés par une équipe dirigée par J. Holland en 1975 à l'Université du Michigan. Ses premiers travaux sont présentés pour la première fois dans son livre « Adaptation in Natural and Artificial Systems ». Ses recherches avaient deux objectifs principaux :

- Mettre en évidence et expliquer rigoureusement les processus d'adaptation des systèmes naturels.

- Concevoir des systèmes artificiels c'est-à-dire des logiciels, qui possèdent les propriétés importantes des systèmes naturels.

Cette approche a débouché à partir de 1983, avec la thèse de D. Goldberg, sur des découvertes importantes à la fois dans les sciences des systèmes naturels et dans celles des systèmes artificiels. L'équipe de D. Goldberg a construit un système artificiel qui s'appuie sur les principes de sélection de C. Darwin [1] et sur les méthodes de combinaison des gènes de J. Mendel [2]. Les recherches dans le domaine des algorithmes génétiques ont pour objectif principal l'amélioration de la robustesse, l'équilibre entre

[1] Charles Darwin (1809-1882) a formulé les principes généraux de la génétique qui sont encore admis aujourd'hui.

[2] Aussi connu comme le moine Gregor Mendel (1822-1884), il a ouvert véritablement la voie de la génétique moderne.

les performances et le coût nécessaire à la survie dans des environnements nombreux et différents.

La programmation génétique se distingue des algorithmes génétiques principalement en ce qui concerne la représentation des individus. En effet, les structures ne sont plus linéaires (comme c'est le cas de celles des algorithmes génétiques) et sont de taille variable, et représentées sous la forme de structures arborescentes. Les opérateurs génétiques de croisement et de mutation sont donc plus complexes en raison de cette représentation. En effet, ce type de structure pose des problèmes particuliers comme le phénomène de « bloat » qui est une augmentation incontrôlée et inutile de la taille des solutions de génération en génération.

4.1.2 Définition et terminologie

Cette section présente le parallèle entre la biologie et les algorithmes génétiques au travers de la définition de ces derniers ainsi que de la terminologie commune à ces deux domaines. Elle décrit les différentes composantes de ces algorithmes.

Définition

Un algorithme génétique travaille sur une population de solutions [3] potentielles. Le processus conduit à l'élimination des individus les plus faibles pour favoriser la survie des individus les plus performants, c'est-à-dire les mieux adaptés. Voici une définition « générale » des algorithmes génétiques :

Les algorithmes génétiques renvoient aux principaux mécanismes de la sélection naturelle, c'est-à-dire essentiellement la sélection, le croisement et la mutation. Un algorithme génétique décrit l'évolution, au cours de générations successives, d'une population d'individus en réponse à leur environnement. Il sélectionne les individus, en accord avec le principe de la survie du plus adapté. Comme leurs équivalents biologiques, les individus sont constitués d'un ensemble de gènes qui ont chacun un rôle propre. Dans une simulation génétique (cf. figure 4.1), les individus les plus adaptés ont une probabilité plus élevée d'être sélectionnés et reproduits, donc d'être présents à la génération suivante. L'opérateur de croisement permet de parcourir l'espace de recherche quant à l'opérateur de mutation d'un gène, celui-ci permet de maintenir une certaine diversité au sein de la population.

Terminologie

Il est d'usage d'utiliser l'expression suivante : *« les algorithmes génétiques trouvent leurs racines à la fois dans la biologie et dans l'informatique »*. De fait, il en découle une terminologie qui est un mélange « de naturel » et « d'artificiel ». Ainsi, les chaînes

[3] Egalement dénommées « individus » ou « chromosomes ».

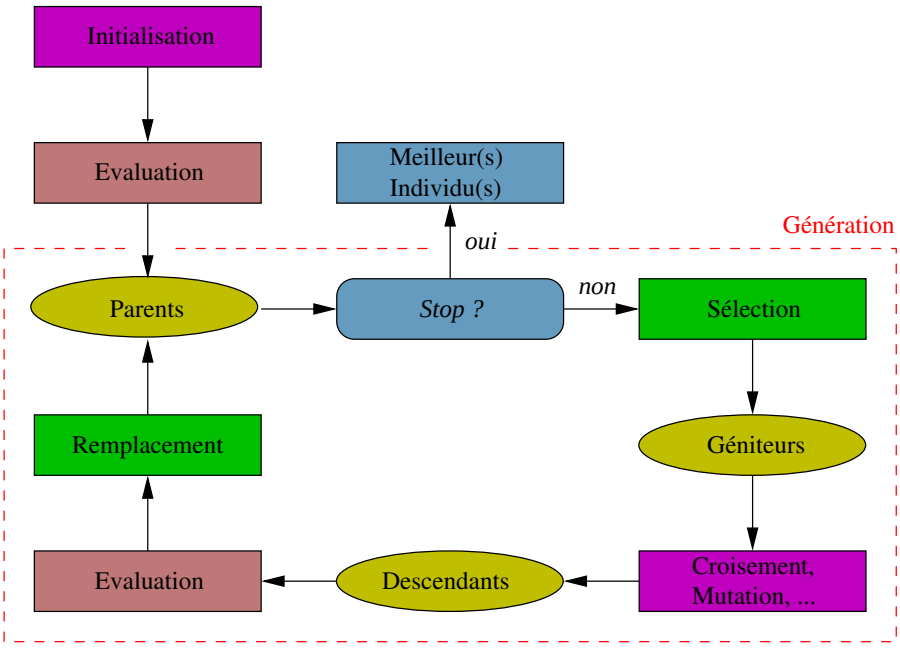


FIG. 4.1 – *Fonctionnement d'un algorithme génétique [eo]. Le choix des couleurs est le suivant : le vert clair pour les génotypes (parents, géniteurs et descendants), le violet pour les opérateurs stochastiques (initialisation, croisement, mutation, ...), le vert foncé pour les opérateurs Darwinistes (sélection et remplacement), le marron pour l'entité la plus coûteuse de l'algorithme : l'application de la fonction d'adaptation. Enfin, le bleu pour les étapes diverses (test des critères d'arrêt, interaction avec l'utilisateur, obtention des solutions finales, obtention de statistiques...)*

(symboles) des systèmes génétiques artificiels sont l'analogue des chromosomes des systèmes biologiques. Dans les systèmes naturels (organismes vivants), un ou plusieurs chromosomes forment ensemble le plan génétique global pour la construction et le fonctionnement d'un organisme. Dans ces systèmes, l'ensemble du matériel génétique est appelé le génotype [4]. Dans les systèmes génétiques artificiels, l'ensemble des chaînes est appelé une structure alors qu'en biologie, l'ensemble des chromosomes s'appelle le génotype. Dans les systèmes naturels, l'organisme formé par l'interprétation de l'ensemble du matériel génétique avec son environnement est appelé le phénotype [5]. Dans les systèmes génétiques artificiels, les structures décodées forment un ensemble de pa-

[4] Le génotype est aussi dénommé « génome ». Il s'agit de l'ensemble de l'information génétique d'une espèce donnée qui est inscrite dans son ADN *(acide désoxyribonucléique)* et qui est transmise d'une génération à l'autre, une moitié provenant du parent mâle et l'autre moitié du parent femelle, lors de la reproduction sexuée.

[5] Cet ensemble de caractères morphologiques permet de reconnaître un individu d'une espèce à une autre à son aspect. Par exemple la couleur des yeux chez l'homme, la taille des nageoires chez les mammifères marins...

ramètres donnés, une solution ou un point dans l'espace des solutions. Dans de tels systèmes, il y a plusieurs manières de coder les paramètres numériques et non-numériques.

Dans la terminologie biologique, on dit que les chromosomes sont constitués de gènes situés linéairement le long de ces chromosomes ; ces gènes peuvent prendre différentes valeurs ou caractères que l'on appelle allèles. En génétique, la position d'un gène (son locus) est identifiée indépendamment de sa fonction. Dans l'exploration génétique artificielle, on dit que les chaînes se composent de traits ou détecteurs, qui prennent différentes valeurs. La correspondance entre la terminologie naturelle et artificielle est résumée dans le tableau 4.1. Il est à noter qu'il est d'usage de considérer un « individu » et un « chromosome » comme étant les mêmes entités, bien que, dans la nature, cela ne soit pas le cas.

Biologie	*Algorithmes génétiques*
chromosome	chaîne
gène	trait caractéristique ou détecteur
allèle	valeur de caractéristique
locus	position dans la chaîne
génotype	structure
phénotype	expression d'un gène

TAB. 4.1 – *Correspondance entre la terminologie de la biologie et celle des algorithmes génétiques.*

L'emploi des algorithmes génétiques se fait systématiquement par le choix d'un codage des individus de la population, par la construction d'un ensemble d'opérateurs génétiques comprenant au moins les opérateurs de sélection, de croisement et de mutation, et par la définition de la fonction d'adaptation [6]. Les trois sections suivantes décrivent les codages, les opérateurs génétiques et les fonctions d'adaptation couramment utilisés dans les algorithmes génétiques.

4.1.3 Représentation des individus

Les individus sont codés le plus souvent sous la forme de chaînes de caractères, ou de chaînes de bits. Différentes techniques sont utilisées dans le but de manipuler des individus de taille variable. La plus simple étant l'utilisation d'individus à N gènes, chaque gène étant associé à un bit spécifiant si celui-ci est utilisé ou pas [Carv00a]. D'autres utilisent des représentations arborescentes, par exemple dans la manipulation

[6] Egalement appelée « fonction de fitness », « fonction d'évaluation » ou encore « fonction d'ajustement ».

de prédicats d'arités différentes [Augi95], mais dans ce cas, l'algorithme génétique doit utiliser des informations auxiliaires, comme certaines connaissances du contexte, lui permettant de manipuler de façon adéquate ces prédicats et les valeurs de ces prédicats.

4.1.4 Opérateurs génétiques

Comme abordé dans la définition de l'algorithme génétique dans la section 4.1.2, la plupart des algorithmes génétiques mettent en œuvre trois opérateurs non-déterministes sur les individus de leur population : la sélection, le croisement et la mutation. Voici plus en détails les principes de chacun de ces opérateurs.

La sélection

La sélection consiste à copier chaque chaîne en fonction des valeurs de la fonction d'adaptation de l'algorithme génétique. Cette fonction peut être envisagée comme une mesure de profit, d'utilité ou de qualité de la population d'individus. Copier des chaînes en fonction des valeurs de leur fonction d'adaptation revient à donner aux chaînes dont la valeur de fonction est plus « grande », une probabilité plus élevée de contribuer à la génération suivante, en créant au moins un descendant. C'est une version artificielle de la sélection naturelle.

Pour réaliser la sélection d'individus, on utilise souvent la technique de « pseudo-hasard » qui emprunte le principe de « la roue de la fortune ». Est accordé dans cette « roue », un secteur plus ou moins grand selon le résultat de la fonction d'adaptation. La génération suivante se détermine en fonction de la mesure s_i des secteurs de la roue calculée selon la formule suivante :

$$s_i = \frac{2\pi \times f_i}{\sum_{k=1}^{N} f_k}$$

avec f_i la valeur de la fonction d'adaptation de l'individu i et N le nombre d'individus dans la population.

Une autre manière assez courante et très efficace de reproduire les individus est d'utiliser la technique du tournoi entre plusieurs « bons » individus (tournoi avec élitisme) pour permettre aux meilleurs de se reproduire, c'est-à-dire d'être présents en un ou plusieurs exemplaires à la génération suivante et ainsi de permettre la transimission de leur matériel génétique dans le temps. L'élitisme évite le danger d'une éventuelle perte du meilleur individu de la génération courante.

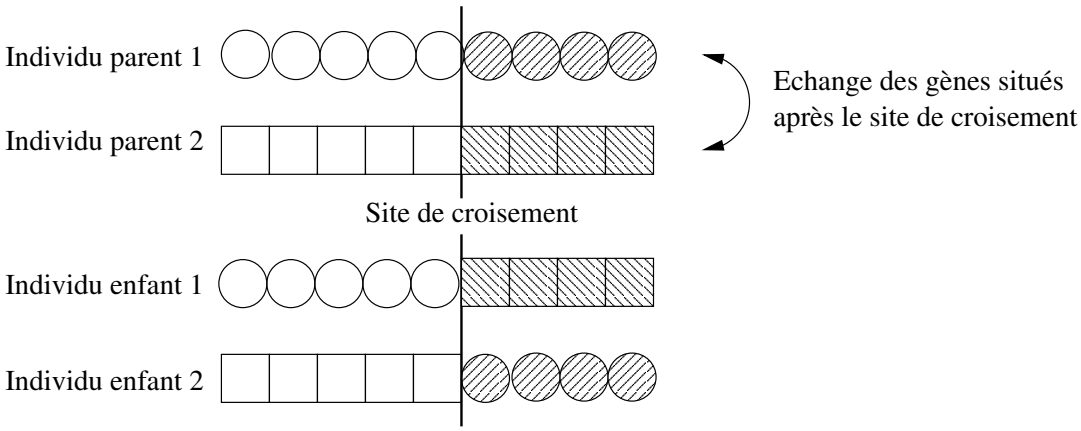


FIG. 4.2 – *Fonctionnement du croisement en un site de deux individus.*

Le croisement

Dans la plupart des applications mettant en oeuvre des algorithmes génétiques, l'opérateur de croisement est appliqué juste après celui de sélection. On distingue deux grandes étapes :

- Premièrement les individus nouvellement obtenus par la sélection sont appariés.

- Deuxièmement, chaque paire d'individus subit un croisement comme suit : un entier k représentant une position sur la chaîne est choisi aléatoirement entre 1 et la longueur l du génotype (1 étant la position du premier gène de cet individu). Deux nouvelles chaînes sont créées en échangeant tous les gènes compris entre les positions $k+1$ et l incluse, ce qui est représenté sur la figure 4.2.

L'opérateur de croisement est utilisé avec une probabilité toujours assez élevée dans le but de diversifier la population.

Ce sont les opérateurs de sélection et de croisement qui donnent toute leur puissance aux algorithmes génétiques. Cependant, bien que ces deux opérateurs explorent et recombinent efficacement les solutions existantes, ils peuvent parfois faire perdre du matériel génétique potentiellement utile, et ceci justifie la nécessité d'un troisième opérateur (opérateur de mutation).

La mutation

Dans un algorithme génétique simple, la mutation est la modification aléatoire occasionnelle (c'est-à-dire de faible probabilité) de la valeur d'un gène de la chaîne. Prise isolément, la mutation constitue une exploration aléatoire de l'espace des chaînes,

mais utilisée de façon « combinée et fine » avec les opérateurs de sélection et de croisement, elle représente un moyen d'éviter la perte d'informations importantes. La probabilité de mutation est souvent très faible dans le but de ne pas altérer les meilleurs individus. Il est d'usage de la définir comme l'inverse de la longueur du génotype.

4.1.5 Fonction d'adaptation

La fonction d'adaptation évalue la « justesse » (fitness) des individus. Elle est en général élaborée de manière empirique. Dans la majorité des cas, plus la valeur de la fonction d'adaptation appliquée à un individu est élevée, meilleur est l'individu considéré. Suivant le contexte, la qualité d'un individu n'a pas le même sens. Il n'existe pas une forme type de fonction d'adaptation mais généralement, chaque « critère » à mesurer sur un individu se modélise dans la fonction par un terme, la fonction d'adaptation étant une combinaison de ces termes.

4.1.6 Justification de l'emploi des algorithmes évolutionnaires

Vastes espaces de recherche

La principale raison justifiant l'emploi d'algorithmes évolutionnaires découle de leur capacité d'exploration de très vastes espaces de recherche. En effet, ces algorithmes travaillent sur un ensemble de points, au lieu d'un point unique. Certaines méthodes d'optimisation « se déplacent » d'un point unique à un autre dans l'espace de recherche en utilisant une règle de transition pour déterminer le nouveau point, ce qui comporte un risque, celui de trouver et de se concentrer sur un optimum local dans des espaces de recherche multimodaux. En revanche, les algorithmes génétiques « escaladent » plusieurs optima locaux de façon simultanée, réduisant ainsi la probabilité de trouver un « mauvais » optimum local.

Formalisme de haut niveau

Ces algorithmes utilisent un codage des paramètres et non les paramètres eux-mêmes. Les algorithmes évolutionnaires et en particulier les programmes génétiques permettent d'utiliser un formalisme de haut niveau pour exprimer les solutions potentielles au travers d'individus. L'emploi de la programmation génétique permet de manipuler des individus représentant des arbres ou de manipuler des expressions mathématiques.

Données corrélées

Contrairement à d'autres méthodes comme la méthode du Gradient, les algorithmes génétiques ne nécessitent pas d'informations auxiliaires pour effectuer leur recherche et déterminer des optima locaux (comme par exemple la donnée de dérivées calculées analytiquement ou numériquement). Ils effectuent leur recherche « en aveugle », ce qui

fait d'eux une méthode d'optimisation plus générale dans le sens où ils n'ont besoin de connaître que des valeurs de la fonction à optimiser associées à chaque individu de la population.

Parallélisme

Les algorithmes évolutionnaires permettent d'exploiter le parallélisme le plus souvent par un partage de la population globale en sous-populations chacune assignée à un processeur. Dans [Alba02], les auteurs présentent les avantages et inconvénients liés aux algorithmes évolutionnaires parallèles.

Solutions hybrides

Une solution est dite hybride lorsqu'elle met en œuvre un algorithme évolutionnaire ainsi qu'un autre algorithme, tel que C4.5 par exemple. Certains travaux [Vent96, Carv99] présentent l'avantage qu'apporte l'utilisation de solutions hybrides basées sur des algorithmes évolutionnaires. Ils mettent en évidence notamment une amélioration des résultats par rapport aux solutions n'utilisant pas ce type d'algorithmes.

Multi-critères

Les algorithmes évolutionnaires peuvent aisément être employés à des fins d'optimisation multi-critères qui consiste à optimiser non plus un critère par le biais de la fonction d'adaptation mais plusieurs fonctions ou critères.

Les cinq points précédents justifient l'emploi des algorithmes évolutionnaires en fouille de données. En effet :

- Dans ce domaine les espaces de recherche sont très vastes en raison du nombre et de la nature des attributs constituant les jeux de données. Les travaux suivants utilisant des algorithmes évolutionnaires en fouille de données mettent en avant cet aspect [Boja99, Frei99a, Noda99, Boja00, Catt00, Cong00, Fide00, Teus00].

- Les algorithmes génétiques, bien qu'offrant un codage des individus plus limité que les programmes génétiques, permettent cependant d'exprimer des règles de manière suffisamment compréhensibles pour l'utilisateur [Arau99, Frei99a, Arau00, Carv00a, Carv00b, Catt00, Cong00, Teus00].

- Les algorithmes évolutionnaires ont de fait la capacité de pouvoir manipuler des données fortement corrélées ; les règles rares [Carv00a, Carv00b] sont très souvent des règles dont les attributs sont fortement corrélés.

- En fouille de données, il est possible de partager l'ensemble de données entre différents processeurs [Arau99, Arau00].

La section suivante présente les travaux en fouille de données relatifs à la mise en œuvre d'algorithmes évolutionnaires basés pour la plupart sur l'emploi d'une seule mesure comme fonction d'adaptation, l'aspect multi-objectifs faisant l'objet du chapitre suivant.

4.2 Algorithmes évolutionnaires en fouille de données

La plupart des travaux actuels relatifs à l'emploi d'algorithmes évolutionnaires en fouille de données portent en grande partie sur la mise en œuvre d'algorithmes génétiques ou de programmes génétiques dans les deux tâches suivantes : en extraction de règles de classification et en recherche de règles d'association. D'autres travaux se rapportent à la découverte de formules logiques générales souvent exprimées en logique du premier ordre. Cette section présente les résultats de recherches récentes relatives à l'utilisation d'algorithmes évolutionnaires dans ces tâches de la fouille de données.

4.2.1 Evaluation globale : attributs corrélés

Les corrélations pouvant exister entre certains attributs dans un jeu de données constituent un véritable problème pour les algorithmes d'induction ; ce problème est d'autant plus important que le nombre d'attributs du jeu de données est élevé [Scha93]. La plupart de ces algorithmes sont qualifiés « d'algorithmes gloutons » en raison de leur manière de parcourir l'espace de recherche, les rendant très peu adaptés à la recherche de règles relatives à des données corrélées contrairement aux algorithmes évolutionnaires comme nous allons le voir dans le paragraphe qui suit. Si l'on considère les algorithmes d'induction d'arbres de décision, ceux-ci choisissent l'attribut du jeu de données le plus discriminant à chaque niveau de l'arbre. La racine « contient » l'ensemble des instances du jeu de données. Ses fils contiennent l'ensemble de ces instances partitionnées selon la valeur d'un des attributs. Chacun de ces fils divise à son tour l'ensemble des instances qu'il contient dans l'ensemble de ses fils et ainsi de suite, à chaque fois selon l'un des attributs. De fait, s'il existe des corrélations entre attributs du jeu de données, cette heuristique ne permet pas de les prendre en compte étant donné que le critère de sélection des attributs est leur « pouvoir discriminatoire » des instances.

Les algorithmes gloutons. Parmi les méthodes de recherche heuristiques, certaines sont connues pour donner en moyenne de bonnes performances. Elles produisent des solutions presque optimales assez rapidement et dans de nombreux cas, elles localisent un optimum local. Les méthodes de recherche locales, par exemple, sont ainsi nommées parce qu'à partir d'un modèle M, elles cherchent un autre modèle parmi ceux qui sont

dans son voisinage. Les méthodes les plus fréquemment employées commencent par le choix d'un point dans l'espace de recherche ; ce choix peut être aléatoire ou dirigé. La recherche procède ensuite en opérant à chaque itération, un déplacement par rapport au point courant en modifiant celui-ci. La qualité du nouveau point est évaluée et la décision de remplacer le point courant par le nouveau point est prise si celui-ci est meilleur. Ces techniques qualifiées de recherche gloutonne, reposent le plus souvent sur l'itération d'étapes de recherche locale dans le voisinage de l'état courant. La qualité de la solution produite est ainsi dépendante du point de départ de l'algorithme. Ces méthodes conduisent le plus souvent à trouver uniquement un optimum local.
Par exemple, l'induction d'un arbre de décision par un algorithme comme C4.5 met en œuvre ce type de recherche gloutonne. Après avoir choisi une fonction d'évaluation adéquate, le gain d'information par exemple, ces algorithmes suivent souvent une méthode heuristique du type « général vers spécifique » : l'algorithme débute avec la forme (règle) la plus générale possible, puis spécialise cette forme à chaque étape, explorant ainsi le voisinage de la règle. Ces algorithmes sélectionnent un attribut à la fois pour constuire une condition de règle. Cependant, il est nécessaire de connaître de façon simultanée les valeurs de certains attributs de prédiction lorsque ceux-ci sont corrélés, car le cas échéant, un autre attribut, qui sera, lui, le plus souvent inadapté, sera choisi à la place.

Les algorithmes évolutionnaires. Ils permettent de prendre en compte les interactions d'attributs qui sont cachées dans les données. En effet, en fouille de données, ces algorithmes travaillent sur des populations d'individus dont chacun représente une règle qui est évaluée en tant que tout. Plusieurs travaux relatifs à l'emploi d'algorithmes évolutionnaires pour prendre en compte l'interaction entre attributs ont été menés. Ainsi, dans [Frei01], l'auteur définit le concept d'interaction entre attributs et met en évidence l'étendue de ce problème sur plusieurs jeux de données. Dans [Dhar00] les auteurs présentent plusieurs caractéristiques de jeux de données financières qui, de par leur nature, rendent la prédiction extrêmement difficile. Une de ces caractéristiques est l'existence d'interactions subtiles et cachées qui ne sont pratiquement jamais détectées par une analyse manuelle ou bien par les algorithmes d'induction standards. Ce dernier travail rapporte également les résultats d'expérimentations relatifs à la comparaison d'un algorithme génétique avec deux algorithmes gloutons d'induction de règles, et présentent les avantages des algorithmes génétiques quant à leur réelle capacité à trouver des interactions cachées entre attributs.
Un autre problème réside dans la présence de règles rares dans les jeux de données, présence souvent causée par l'interaction entre attributs. En effet, dans de tels jeux de données, il est nécessaire de prendre en compte de façon simultanée les valeurs de plusieurs attributs de prédiction ce qui augmente la taille des règles et de fait, diminue le nombre d'instances que leur antécédent satisfait. Ces règles sont donc par définition des règles rares. Ce phénomène, présent dans plusieurs jeux de données [Weis00a, Carv01, Carv02a] est exposé en détails au chapitre 3.

Enfin des auteurs mettent en évidence que les algorithmes génétiques surpassent les algorithmes gloutons lorsqu'ils sont appliqués à des jeux de données artificiels comportant un haut degré d'interaction entre attributs [Papa01].

4.2.2 Parcours de l'espace de recherche

Fonction d'adaptation

Il est souvent énoncé le principe selon lequel les connaissances découvertes par les algorithmes de fouille de données devraient satisfaire trois principaux critères qui sont *la précision*, *la compréhensibilité* et *l'intérêt*. Nous avons présenté dans le chapitre 3 un certain nombre de mesures de qualité et parmi celles-ci nous pouvons citer comme étant des mesures de précision : le $Support$, la $Confiance$, la $Sensibilité$, la $Spécificité$ et le $Taux\ de\ succès$. La compréhensibilité d'un modèle lorsque celui-ci est constitué de règles est plutôt relative à leur forme ou bien à la longueur des règles. L'intérêt quant à lui revêt plusieurs aspects qui peuvent être le pouvoir à surprendre l'utilisateur, l'aptitude à contredire des connaissances à priori du domaine...

Précision. Nous présentons ici la manière dont ces critères sont pris en compte, de façon individuelle ou simultanée dans les travaux existants relatifs aux algorithmes évolutionnaires en fouille de données et pour la recherche de règles.
Nous avons présenté dans le chapitre 3 un certain nombre de mesures dont plusieurs sont employées dans des algorithmes évolutionnaires. Pour ce faire, la fonction d'adaptation peut être basée sur une ou plusieurs de celles-ci sous la forme d'une expression mathématique qui les combine.
Un des intérêts de la fonction d'adaptation est de permettre la combinaison de plusieurs mesures. Dans [Weis99], l'auteur utilise la fonction d'adaptation f définie par :

$$f(r) = Confiance(r) \times Sensibilité(r)$$

Cette fonction est utilisée pour la découverte de règles rares. L'idée étant d'optimiser de façon simultanée la $Confiance$ et la $Sensiblité$ des règles par le produit. En effet, une règle rare a généralement une $Confiance$ élevée et optimiser sa $Sensibilité$ permet d'écarter les règles triviales. Il est également possible d'utiliser une fonction d'adaptation basée sur la $Sensiblité$ et la $Specificité$ de la façon suivante :

$$f(r) = Sensibilité(r) \times Spécificité(r)$$

Par exemples, les travaux suivants sont basés sur cette fonction [Carv00a, Carv00b, Fide00, Carv02b].

Parfois il est nécessaire d'utiliser des fonctions d'adaptation qui dépendent du domaine. Celles-ci doivent alors prendre en compte un certain nombre de mesures permettant de fournir des informations en adéquation avec les attentes de l'utilisateur. On peut par exemple citer les travaux de [Thom99, Thom00] dans lesquels il s'agit d'utiliser un programme génétique dans le domaine de la finance pour maximiser le profit de règles de transactions, au lieu de maximiser directement la précision des règles. Comme nous l'avons évoqué, la précision ne constitue qu'un aspect de la qualité globale d'une règle.

Compréhensibilité. La compréhensibilité d'une règle est relative à sa longueur ou à sa syntaxe. Ainsi, en général, on considère que plus une règle est courte, plus elle est compréhensible. Cet aspect peut être aisément incorporé à l'algorithme évolutionnaire par le biais de sa fonction d'adaptation. En général ceci est réalisé au moyen d'une fonction pondérée, avec un terme mesurant la précision de la règle et un autre terme mesurant la longueur, chaque terme étant assigné d'un poids prédéfini. Plusieurs travaux sont relatifs à cette méthode [Jani93, Pei97]. Cette approche est facile à implémenter mais ignore cependant l'aspect subjectif de la compréhensibilité. L'alternative consiste à utiliser une fonction d'adaptation interactive pour prendre ce point en compte [Banz00].
L'autre aspect relatif à la compréhensibilité d'une règle est lié à sa syntaxe. En effet, l'utilisateur peut avoir des préférence pour certains attributs. Ainsi, favoriser au sein de l'algorithme évolutionnaire les règles basées sur ces attributs, permet la prise en compte de ce critère. Celui-ci peut être implémenté dans la fonction d'adaptation par l'utilisation de coefficients associés aux attributs préférentiels de façon à réduire les chances de survie des règles ne comportant pas ces derniers.

Intérêt. Enfin, considérons le dernier critère devant être considéré par une fonction dadaptation : l'intérêt des règles. Comme nous l'avons présenté dans la section 3.1.3, il existe deux approches quant à l'intèrêt d'une règle : l'approche *objective* et l'approche *subjective*. L'approche subjective peut consister en un processus interactif afin de prendre en compte « l'avis » de l'utilisateur. Nous nous intéressons plus particulièrement à la première catégorie de mesures d'intérêt : $Lift$, $Rule\ Interest$, $\mathcal{J}Mesure$, $Sebag$ et $Conviction$. Bien entendu, cette approche consiste à baser la fonction d'adaptation de l'algorithme évolutionnaire sur une ou plusieurs mesures d'intérêt. Par exemple, on peut considérer une fonction constituée d'un terme mesurant la précision de la règle et un autre mesurant son intérêt sous la forme d'une combinaison de termes où chaque terme est assigné d'un poids prédéfini. Cette approche est utilisée par [Noda99] dans un algorithme génétique spécialement conçu pour la découverte de règles de prédiction « intéressantes ». Une mesure d'intérêt peut également être utilisée comme fonction d'adaptation à part entière et ce parmi un éventail de mesures

que l'utilisateur sélectionne selon ses objectifs. C'est le cas dans l'algorithme génétique développé dans [Floc95].

Le croisement

Dans le souci d'obtenir des résultats cohérents (des règles compréhensibles et courtes par exemple) il est nécessaire que cet opérateur évite de dupliquer du matériel génétique lors de la production des descendants [Arau99, Cong00]. Du matériel génétique invalide est rapidement obtenu avec le croisement classique, c'est pourquoi dans la plupart des cas, cet opérateur doit vérifier qu'il ne produit pas d'incompatibilité au niveau des individus produits (par exemple, duplication de termes faisant intervenir le même attribut et la même valeur mais des opérateurs logiques différents et contradictoires).

La mutation

Dans l'utilisation des algorithmes génétiques en fouille de données, il existe différentes mutations suivant le problème considéré. On distingue la mutation d'un attribut, dans ce cas l'opérateur doit s'assurer que la valeur de l'ancien attribut est compatible avec le nouvel attribut et la mutation d'un terme dans laquelle à la fois l'attribut, la valeur et éventuellement l'opérateur liant l'attribut à sa valeur sont générés, le plus souvent aléatoirement [Arau99, Frei99a]. La mutation peut permettre dans certains cas de supprimer des termes dans le but de produire des règles courtes donc plus compréhensibles [Arau99]. Certains travaux concernant la manipulation de prédicats au travers des individus introduisent un opérateur de mutation permettant de réaliser des généralisations de prédicats conformément à une structure hiérarchique connue de l'algorithme génétique. Une telle généralisation peut être également appliquée à des symboles, par exemple pour la généralisation d'une constante en variable [Augi95].

4.2.3 Optimisation multi-objectifs

Dans la section précédente, nous avons présenté une approche consistant à prendre en compte de façon simultanée plusieurs mesures au sein de la fonction d'adaptation. Néanmoins, bien que cette méthode soit populaire dans la littérature, elle ne constitue par le moyen idéal de prendre en compte des mesures qui présentent des antagonismes [Fons00, Deb01]. Or, les mesures de précision, de compréhensibilité et d'intérêt peuvent être « conflictuelles » dans de nombreux cas. Cet aspect constitue la nécessité de mettre en œuvre une méthode multi-objectifs que nous présentons au chapitre 5.

4.2.4 Codage des individus

Dans un algorithme génétique dont les individus représentent des règles comme c'est le cas en recherche de règle de dépendance, il existe deux approches pour coder les individus de la population : l'approche de *Michigan* et l'approche de *Pittsburgh*.

Voici dans le cadre de la recherche de règles de dépendance le détail de chacune de ces approches :

- **L'approche de Michigan** : un individu représente une unique règle. Dans cette approche, il y a deux possibilités pour découvrir des règles. La première qui est la plus simple, consiste à exécuter l'algorithme génétique pour découvrir une seule règle. Ainsi, pour découvrir un ensemble de règles, il est nécessaire de réaliser plusieurs exécutions. Il en découle un inconvénient qui est une durée importante de traitement. La seconde possibilité consiste à faire découvrir un ensemble de règles à l'algorithme génétique. Nous avons choisi cette approche dans les expérimentations qui suivent. La principale difficulté associée à l'approche de Michigan, dans sa seconde forme, est de trouver l'ensemble des meilleures règles. En effet, l'ensemble des meilleures règles n'est pas le meilleur ensemble de règles car il peut exister des interactions entre les règles. Pour prendre en compte cette interaction, il est nécessaire de mettre en œuvre un mécanisme de niches. Nous introduisons le concept de niches écologiques au chapitre 5.

- **L'approche de Pittsburgh** : un individu est composé d'un ensemble de règles. Cette approche prend tout naturellement en compte l'interaction entre règles puisqu'un individu est un ensemble de règles, et qu'un individu est évalué comme un tout.

Il existe plusieurs manières de coder l'antécédent et le conséquent d'une règle :

- **Codage de l'antécédent** : chaque gène correspond à un attribut et représente une condition composée d'un opérateur et d'une valeur. Si le codage binaire est choisi, plusieurs cas apparaissent : lorsque l'attribut est binaire ou catégoriel, il suffit d'associer une chaîne de bits à chacune des valeurs. La démarche est la même pour l'opérateur. Lorsque l'attribut est continu, une première approche consiste à discrétiser les valeurs de son domaine afin de retomber dans le cas d'un attribut catégoriel. Cependant, cette étape de discrétisation est un pré-traitement effectué par des algorithmes de discrétisation, algorithmes généralement reconnus comme ne prenant pas en compte l'interaction entre attributs. Ceci constitue un problème car l'algorithme génétique sera alors dans l'incapacité de mettre en évidence d'éventuelles interactions entre les attributs du jeu de données. Heureusement, les algorithmes génétiques peuvent se passer d'un codage binaire et utiliser directement au sein des individus un codage réel pour les attributs réels. L'avantage du codage binaire est la simplicité des opérateurs génétiques associés. Son inconvénient est l'acroissement de la longueur des individus avec celui des domaines des attributs en présence. L'avantage du codage « direct » est une plus grande facilité de représentation et l'absence de pré-traitement. Cependant, avec cette approche, les opérateurs génétiques sont plus complexes. Les opérateurs

utilisés dans les conditions sont quant à eux le plus souvent les opérateurs $<$, $\leq$, $=$, $\geq$ et $>$.

- **Codage du conséquent** : la première approche consiste à coder le conséquent directement dans le génotype d'un individu. En recherche de règles de dépendance simples comme les règles de classification, il n'est pas nécessaire de coder le nom de l'attribut étant donné qu'il est le même pour tous les individus et donc celui-ci peut être fixé. En recherche de règles ayant plus d'un attribut dans le conséquent [7], un gène du conséquent doit coder également le nom de l'attribut correspondant étant donné que le conséquent est variable. Il existe trois approches pour choisir le conséquent d'une règle. La première, que nous avons présentée, consiste à fixer celui-ci pour toutes les règles. Cette manière est adaptée à la recherche de règles de classification. En revanche, il est souhaitable d'effectuer ce choix du conséquent « à la volée » c'est-à-dire pendant l'exécution de l'algorithme génétique, ce qui est le cas en recherche de règles généralisées. La dernière approche consiste à fractionner la population en sous-populations, chacune assignée d'une valeur pour le conséquent (règles simples) ou bien d'un conséquent particulier (règles généralisées).

Un autre aspect important lié à la représentation des individus, est qu'il est parfois nécessaire de coder des règles de longueur variable. Ce que nous allons présenter s'applique aussi bien au codage de l'antécédent, qu'au codage du conséquent des règles généralisées. Il existe deux approches. Soit les gènes sont de taille fixe, et la longueur du génotype est fixée. Dans ce cas, certains gènes sont rendus « inactifs » par l'algorithme génétique au cours de son exécution afin d'obtenir des règles de longueurs variables. Soit, un individu est codé sous la forme d'une liste de gènes qui correpond directement à une règle. Ces deux codages supposent un codage de positions des gènes dans le génotype. La première approche simplifie l'implémentation de l'opérateur de croisement. Dans le cas de règles généralisées, les opérateurs génétiques de croisement et de mutation doivent être conçus de manière à éviter d'engendrer des individus non valides (vides par exemple).

4.2.5 Opérateurs génétiques

Nous présentons ici les dérivés des opérateurs génétiques classiques adaptés aux règles de dépendance. Il s'agit des opérateurs de croisement et de mutation pour la généralisation et la spécialisation de règles [Gior94, Angl97], et des opérateurs d'ajout et de suppression de gènes.

- **Le croisement pour la généralisation** de règles est appliqué lorsque les deux règles correspondant aux deux individus croisés sont trop spécifiques. Cela sous-

[7] « Dependancy modeling » dans la terminologie anglo-saxonne.

entend la présence d'un mécanisme permettant de mesurer à quel point ces règles sont spécifiques, par exemple par l'emploi du *Support*. Il s'agit d'effectuer le croisement de manière classique puis d'effectuer un OU logique entre certains gènes (par exemple entre les gènes situés après le site de croisement s'il n'y en a qu'un, ou bien entre les deux sites de croisements s'il y en a deux...)

- **Le croisement pour la spécialisation** de règles fonctionne de la même manière excepté qu'il est appliqué à des règles qui sont jugées trop générales. Il s'agit alors d'effectuer un ET logique entre certains gènes.

- **La mutation pour la généralisation** de règles se fait à un niveau de granularité inférieur à celui des opérateurs de croisement précédents. Il s'agit de généraliser un ou plusieurs gènes par une autre valeur ou un autre opérateur logique rendant la condition plus générale.

- **La mutation pour la spécialisation** consiste de la même manière que pour l'opérateur précédent, à rendre la condition plus spécifique par l'emploi d'une autre valeur et/ou d'un autre opérateur logique.

- **Le rajout et la suppression de gènes**. Il s'agit d'autres manières de spécialiser et généraliser une règle par l'ajout et la suppresion d'une condition dans la règle. Ce type d'opérateur est associé à un niveau de granularité supérieur aux deux précédents.

4.3 Approche expérimentale

Dans le chapitre précédent, nous avons présenté diverses caractéristiques liées aux espaces de recherche. Dans cette section, nous illustrons l'apport de l'emploi des algorithmes évolutionnaires pour apporter des solutions aux problèmes mis en évidence. Nous présentons l'architecture de l'algorithme génétique que nous mettons en œuvre.

4.3.1 Algorithme

Notre algorithme est implémenté en langage C++ à partir de la bibliothèque EO Evolutionary Computation Framework [eo]. Il s'agit d'un algorithme génétique générique pour la découverte de règles de dépendance. Cet algorithme est générique dans le sens où il peut être appliqué à n'importe quel jeu de données. Les opérateurs de mutation et de croisement sont donc adaptés aux deux types de règles : règles *simples* et règles *généralisées*. Cette section présente les différents constituant de cet algorithme.

Représentation des individus

Un individu, qui correspond à une règle, est représenté par un tableau de taille fixe dans lequel chaque élément, c'est-à-dire chaque gène, est un terme de la forme :

opérateur	*valeur*	*présence*

où :

- *opérateur* peut prendre la valeur $=$ dans le cas où l'attribut correspondant au gène est catégoriel et dans le cas où l'attribut est numérique, celui-ci peut également prendre la valeur $<$, $\leq$, $\geq$ ou encore $>$.

- *valeur* est une des valeurs appartenant au domaine de l'attribut.

- *présence* est un booléen permettant de spécifier si le gène est présent ou pas ce qui permet à l'individu, bien qu'ayant un génotype de taille fixée, de représenter une règle de taille variable : si ce booléen est à « vrai », le terme est considéré dans la règle représentée par l'individu sinon, le terme en est absent.

Il existe une bijection entre l'ensemble des gènes d'un individu et celui des attributs du jeu de données. Ainsi, le $i^{ème}$ gène correspond au $i^{ème}$ attribut. Par conséquent, il est inutile de représenter le nom de l'attribut dans le gène. De part la dichotomie énoncée ci-dessus concernant les types de règles en présence, il existe deux représentations différentes :

- La première consiste à ne pas représenter la partie droite dans l'individu. Dans ce cas, tous les individus ont le même conséquent.

- La seconde consiste à modifier la représentation ci-dessus par l'ajout d'un champ dénommé *conséquent* :

opérateur	*valeur*	*présence*	***conséquent***

  Ce champ est un booléen permettant de spécifier si un terme est un terme de conséquent. Ceci a un impact sur le croisement et la mutation, et dans une moindre mesure, sur l'affichage de la règle.

Initialisation de la population initiale

Nous avons choisi, suite à un certain nombre de tests, d'effectuer l'initialisation d'un individu de la façon suivante : un individu est initialisé soit à partir d'une entrée

du jeu de données dont la position est déterminée de façon aléatoire dans $\{0, ..., N\}$ avec N le nombre d'entrées du jeu, soit en générant chaque gène de façon aléatoire (c'est-à-dire en déterminant un opérateur et une valeur ainsi qu'un état « présent » ou « absent »). Le choix de l'une ou l'autre de ces initialisations étant déterminée avec une probabilité de 1/2. Il s'avère que les résultats sont meilleurs lorsque la population est initialisée ainsi, plutôt qu'en initialisant les individus sans avoir recours au jeu de données. En effet, sélectionner une instance pour générer un individu est une garantie que cette instance (au moins) satisfera l'individu. En effet, tous les gènes ne sont pas rendus actifs lors de cette initialisation, ce qui augmente la probabilité pour l'individu d'être satisfait par plus d'une instance. Dans le cas de l'initialisation aléatoire, par contre, l'algorithme nécessite un certain nombre de générations avant d'engendrer des individus aux qualités similaires, ce qui explique la différence de résultats entre ces deux méthodes (lorsque le nombre de générations est fixé).

Opérateur de croisement

En raison des deux types de représentations des règles, deux types de croisement sont disponibles. En effet, selon que les individus représentent des règles *simples* ou *généralisées*, le croisement n'opère pas de la même manière. Ces croisements sont les suivants :

- Le croisement des règles *simples* détermine un site de croisement de façon aléatoire puis échange les matériels génétiques des deux individus situés après le site.

- Celui des règles généralisées est plus complexe afin de garantir que les individus obtenus à l'issue du croisement soient valides. En effet, prenons l'exemple de la figure 4.3. Il s'agit de deux individus représentant deux règles généralisées. Les gènes correspondant aux termes du conséquent des deux règles sont hachurés. En effet le conséquent de telles règles peut être constitué de n'importe quel ensemble de termes d'une cardinalité non nulle et inférieure au nombre d'attributs du jeu de données ; ainsi dans l'exemple, le conséquent de la première règle est relatif au premier et au quatrième attribut du jeu de données. Quant à la seconde règle, seuls le septième et le dernier terme appartiennent au conséquent de cette règle.

Bien entendu, un autre cas de croisement incorrect concerne celui où l'un des deux individus engendrés ne contient que des termes de conséquent. En effet, si les deux individus parents ont tous leurs gènes de conséquent situés respectivement avant le site de croisement pour l'un et après le site de croisement pour l'autre, alors un individu descendant sera constitué uniquement de gènes de conséquent et l'autre uniquement de gène d'antécédent ce qui n'a pas de sens.
En présence de telles « configurations génétiques », le site de croisement est choisi de manière à éviter la génération d'individus incohérents ; si ce site ne peut être trouvé, le croisement n'a pas lieu.

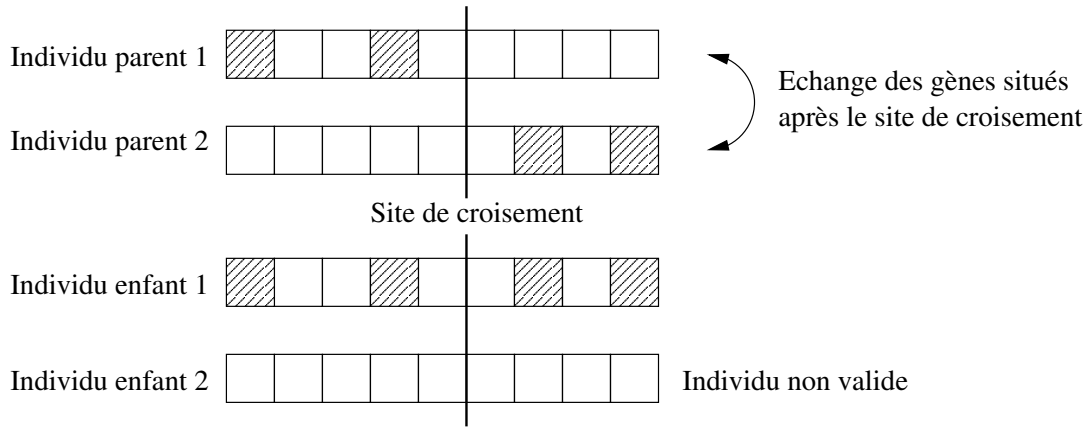


FIG. 4.3 – *Un exemple de croisement générant un individu non valide.*

Opérateur de mutation

Comme pour le croisement, deux opérateurs de mutation existent. Le premier, effectue une mutation des règles simples de la façon suivante : un gène est sélectionné de façon aléatoire avec une probabilité égale à l'inverse de la longueur du génotype. Si le gène de ce locus est inactif c'est-à-dire que le terme correspondant est absent de la règle, celui-ci est rendu actif, et un opérateur logique ainsi qu'une valeur sont générés de façon aléatoire. Si le gène est présent, deux situations se présentent : soit le gène est rendu absent, soit son opérateur logique et sa valeur sont ré-initialisés avec une probabilité de 1/2 dans chaque cas.
Dans le cas des règles généralisées, l'opérateur agit de la même manière excepté qu'il s'assure de ne pas provoquer la génération d'un individu invalide ne contenant aucun gène de conséquent ou que des gènes de conséquent.

Sélection

La sélection consiste en un tournoi à deux individus. Deux individus sont choisis aléatoirement dans la population, celui qui possède la valeur d'adaptation la plus élevée est sélectionné pour le croisement.

Fonction d'adaptation

La fonction d'adaptation consiste soit en une mesure d'intérêt, soit en une combinaison linéaire de mesures selon le choix de l'utilisateur. Dans le cas de combinaisons linéaires, celui-ci peut spécifier un ensemble de n poids pour les n mesures utilisées. L'évaluation d'un individu consiste en un parcours exhaustif des instances du jeu de données sur lequel l'algorithme génétique est exécuté

4.3.2 Mesures de qualité et fonction d'adaptation

Dans cette section, nous étudions les règles selon diverses approches : la première consiste à mettre en évidence de « bonnes » règles comportant de nombreux termes. Ces règles trouvées par l'approche évolutionnaire ne le sont pas par la méthode standard en raison de leur forme. La seconde approche consiste à montrer l'existence de règles éloignées d'une règle de référence fournie par la méthode standard, selon la distance de Hamming.

Longueur des règles

La règle simple suivante qui est relativement longue, est obtenue sur le jeu de données « Vote » avec une fonction d'adaptation réduite à la mesure de $Lift$. La valeur du $Lift$ pour cette règle est 2.50837 c'est-à-dire relativement élevée si l'on en juge l'approximation de l'espace de recherche de cette mesure sur ce jeu, qui est présentée au chapitre 3. En effet, la grande majorité des règles dans ce graphe a une valeur de $Lift$ inférieure à celle de cette règle. D'autre part elle a un $Support$ de 7.13% et une $Confiance$ de 96.88%. Cette règle n'est pas déterminée par l'algorithme d'induction d'arbres de décision dérivé de C4.5 que nous avons mis en œuvre et qui passe par une phase d'élagage. On note d'autre part, que les attributs constituant cette règle ne sont communs à aucune règle produite par l'arbre de décision car ceux-ci ne sont pas assez discriminants selon l'heuristique employée dans cet algorithme. Cette règle est localisée sur le graphe de la figure 4.4.

*« **SI** handicapped-infants=n **ET***
*water-project-cost-sharing=y **ET***
*religious-groups-in-schools=y **ET***
*aid-to-nicaraguan-contras=n **ET***
*immigration=y **ET***
*education-spending=y **ET***
duty-free-exports=n
***ALORS** Class=republican. »*

Sur le même jeu de données, la règle généralisée suivante est trouvée avec une fonction d'adaptation égale au *Taux de succès*. Son $Support$ est de 12.41% et sa $Confiance$ de 58.06%. On peut localiser cette règle sur le graphe de la figure 4.5, son *Taux de succès* vaut 99.01%. On peut observer sur cette figure que cette règle fait partie de celles qui ont les plus longs conséquents d'une part et qui optimisent le *Taux de succès*.

*« **SI** immigration=y **ET***
*crime=y **ET***
Class=republican
ALORS** physician-fee-freeze=y **ET

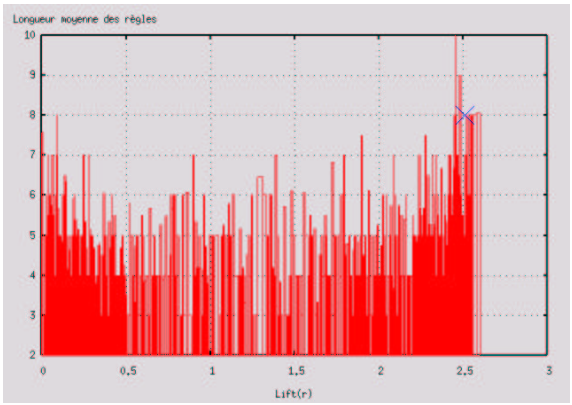


FIG. 4.4 – *Forme des antécédents des règles simples basées sur le Lift et obtenues à partir du jeu de données « Vote ». La croix bleue (en haut à droite) correspond à la règle trouvée par l'algorithme génétique.*

el-salvador-aid=y ***ET***
religious-groups-in-schools=y ***ET***
aid-to-nicaraguan-contras=n ***ET***
mx-missile=n ***ET***
superfund-right-to-sue=y ***ET***
duty-free-exports=n ».

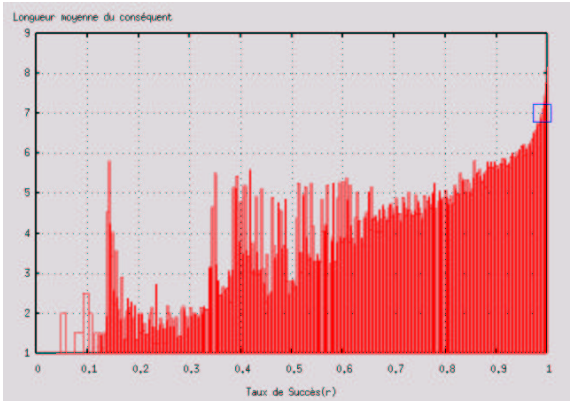


FIG. 4.5 – *Forme des conséquents des règles généralisées basées sur le Taux de succès et obtenues sur le jeu de données « Vote ». Le carré bleu (en haut à droite) correspond à la règle trouvée par l'algorithme génétique.*

Variations

La règle ci-dessous, extraite à partir du jeu de données « Vote », est un exemple de règle trouvée par l'algorithme génétique dont la fonction d'adaptation est réduite à la $Confiance$. Cette règle, à une distance de 7 par rapport à la règle trouvée par l'algorithme d'induction d'arbres de décision, est une des règles n'ayant aucun attribut en commun avec la règle de référence R_0 que nous utilisons. La règle ci-dessous, qui est une règle rare a une $Confiance$ maximale. Il n'est pas surprenant que celle-ci ne soit pas trouvée par l'algorithme d'induction d'arbres de décision, cependant l'algorithme génétique envisage un tout autre ensemble d'attributs pour obtenir les règles du modèles. La règle R suivante est localisée sur le graphe de la figure 4.6.

*R : « **SI** handicapped-infants=n **ET***
*water-project-cost-sharing=y **ET***
*adoption-of-the-budget-resolution=n **ET***
*anti-satellite-test-ban=n **ET***
*immigration=y **ET***
*duty-free-exports=n **ET***
export-administration-act-south-africa=y
***ALORS** Class=republican. »*

Les algorithmes d'induction d'arbres ont tendance à extraire des règles de bonne $Confiance$ en raison du critère qu'ils utilisent et qu'ils optimisent à chaque niveau de l'arbre. Dans le cas présent, ces algorithmes ne trouvent pas la règle R car ils opèrent une recherche locale dans le voisinage de la règle R_0. L'algorithme génétique permet de découvrir cette règle car il effectue un parcours beaucoup plus efficace de l'espace de recherche.

Bien que l'algorithme d'induction d'arbre de décision optimise le $Taux\ de\ succès$ global du classifieur, on peut objecter que le $Taux\ de\ succès$ des règles qui le composent tend également à être optimisé. Comme pour la $Confiance$, l'algorithme génétique basé sur le $Taux\ de\ succès$, trouve des règles aussi bonnes voir meilleures que celles trouvées par l'algorithme d'induction d'arbres de décision. Voici ci-dessous un exemple d'une telle règle R', elle aussi trouvée sur le jeu de données « Vote » (cf. figure 4.7). Sa distance n'est pas très élevée par rapport à la règle de référence R'_0. Le $Support$ de cette règle est de 37.5% et sa $Confiance$ de 92.1%. Cette règle n'est pas trouvée par l'algorithme d'induction d'arbres de décision.

*R' : « **SI** physician-fee-freeze=y **ALORS** republican=y. »*

Combinaison linéaire de mesures de qualité

De manière à évaluer les règles selon plusieurs critères, nous avons mis en œuvre une approche évolutionnaire basée sur une fonction d'adaptation composée de plusieurs

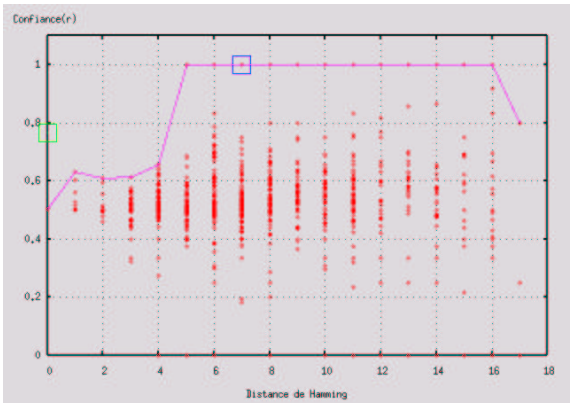


FIG. 4.6 – *Distances de Hamming entre la règle de référence R_0 générée par l'algorithme d'induction d'arbre de décision et les règles générées aléatoirement et évaluées sur la Confiance sur le jeu de données « Abonnés ». Le carré vert (à gauche) correspond à la règle de référence R_0 et le carré bleu (à droite) à la règle R trouvée par l'algorithme génétique.*

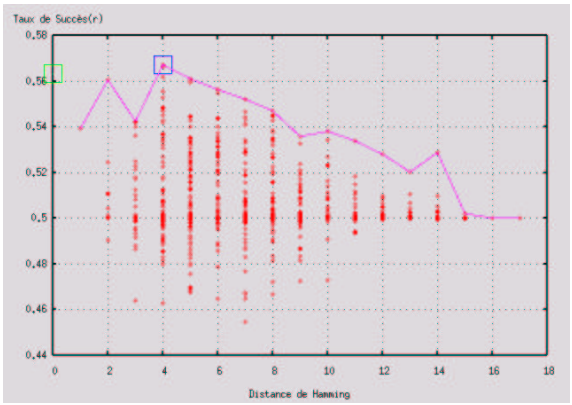


FIG. 4.7 – *Distances de Hamming entre la règle de référence R'_0 générée par l'algorithme d'induction d'arbre de décision et des règles générées aléatoirement et évaluées sur le Taux de succès sur le jeu de données « Abonnés ». Le carré vert (à gauche) correspond à la règle de référence R'_0 et le carré bleu (à droite) à la règle R' trouvée par l'algorithme génétique.*

mesures. Les mesures correspondant à des critères différents comme la longueur et le $Lift$, par exemple, ne sont souvent pas corrélés et sont dans certains cas antagonistes. L'algorithme génétique utilisé met en œuvre un mécanisme de niches écologiques [8] afin que la population finale ne soit pas dominée par un même individu. Par une spécifi-

[8] Nous introduisons ce concept en détails au chapitre 5.

cation des poids de chaque mesure, il est possible d'obtenir un ensemble de points de l'espace de recherche qui satisfont de façon simultanée les diverses mesures considérées. Cependant, comme le fait remarquer l'auteur dans [Frei02], cette approche est simple mais pose problème lorsque les mesures sont antagonistes. En effet, dans ce cas, il est nécessaire d'effectuer une exécution de l'algorithme pour chaque ensemble de poids ce qui généralement conduit à des temps de calcul prohibitifs. En effet, l'exécution peut s'avérer longue et il est difficile de choisir quels poids utiliser si ce n'est par le biais d'une approche « essai-échec ». De plus, une telle approche est sensible à la forme de ce que l'on appelle la surface de compromis et que nous définissons dans le chapitre suivant. L'alternative consiste à mettre en œuvre une méthode multi-objectifs permettant de déterminer dans tous les cas une approximation de l'ensemble de compromis.

4.3.3 Conclusion

Dans ce chapitre nous avons présenté la métaheuristique des algorithmes évolutionnaires et en particulier celle des algorithmes génétiques. Nous avons mis en lumière leurs avantages ainsi que leur adaptation en fouille de données.
Nous avons également présenté l'approche évolutionnaire comme une solution adaptée aux caractéristiques des espaces de recherche des règles évaluées selon différents critères de qualité. Les résultats que nous avons obtenus sur différents jeux de données, nous ont permis de conclure que dans certains cas, les algorithmes génétiques permettent d'identifier des solutions ignorées par des méthodes plus traditionnelles. Notre approche, qui rappelons-le est expérimentale, permet notamment d'apporter des solutions lorsque qu'il existe de « bonnes » règles de dépendance selon une mesure mais qui sont ignorées par les techniques standards car trop longues. D'autre part, certaines règles non envisagées par ces mêmes techniques peuvent également être découvertes. Enfin, nous avons abordé le cas de mesures antagonistes qui nécessitent une approche multi-objectifs pour obtenir des modèles intéressants, ce qui fait l'objet du chapitre suivant.

Chapitre 5

Algorithmes génétiques multi-objectifs en fouille de données

Sommaire

Ce chapitre est consacré à la mise en œuvre d'un algorithme génétique générique dans un contexte multi-objectifs en fouille de données. Le chapitre précédent a présenté notre première approche qui consistait à utiliser une mesure de qualité comme fonction d'adaptation au sein de l'algorithme génétique. Le présent chapitre étend cette idée à la donnée de plusieurs mesures et en particulier de deux. Il est structuré de la façon suivante : la première section est relative à l'optimisation multi-objectifs d'une façon générale. La seconde section présente diverses méthodes relatives à ce domaine. L'accent est mis dans la section suivante sur les méthodes d'optimisation multi-objectifs sur lesquelles notre choix s'est porté à savoir les algorithmes génétiques. Vient ensuite une section situant les travaux actuels relatifs à l'optimisation multi-objectifs mettant en œuvre des algorithmes génétiques en fouille de données. La dernière section présente l'intérêt de cette approche au vu de résultats expérimentaux obtenus selon deux approches : celles relatives à l'analyse des graphes et celles relatives à l'analyse des règles.

5.1 Optimisation multi-objectifs

5.1.1 Introduction

De nombreux secteurs comme l'électronique, la conception de systèmes mécaniques ou encore le traitement de l'image présentent des problèmes technologiques auxquels il est souvent difficile de faire face. Le plus souvent, le problème en question peut s'exprimer sous la forme d'un *problème d'optimisation* dans lequel une ou plusieurs fonctions d'objectif (ou fonctions de coût) sont définies. L'idée étant de minimiser ou de maximiser c'est-à-dire d'optimiser ces fonctions selon un ensemble de paramètres. La définition du problème d'optimisation est souvent complétée par la donnée de *contraintes*, c'est-à-dire que tous les paramètres (ou *variables de décision*) de la solution proposée doivent respecter ces contraintes.
Les méthodes d'optimisation « classiques » pour la résolution de tels problèmes sont nombreuses lorsque des conditions mathématiques sont remplies. Par exemple, lorsque la fonction objectif et les contraintes s'expriment linéairement en fonction des variables de décision, la *programmation linéaire* traite efficacement ce genre de problèmes. Cependant, dans la pratique, ces méthodes sont souvent impuissantes en raison de l'existence d'une ou plusieurs particularités comme par exemple l'impossibilité d'exprimer la fonction objectif selon les variables de décision ou encore la prise en compte simultanée de plusieurs objectifs contradictoires.
On distingue ainsi deux aspects : l'*optimisation multi-objectifs* et l'*optimisation difficile*, qui se concentrent sur deux différents types de particularités. L'optimisation multi-objectifs, qui est le domaine dans lequel nous nous plaçons, traite le cas de la présence simultanée de plusieurs objectifs alors que l'optimisation difficile analyse les difficultés propres aux propriétés mêmes de chaque fonction objectif.
La section suivante présente une définition plus formelle de l'optimisation multi-objectifs.

5.1.2 Définitions

Optimisation multi-objectifs. L'optimisation *multi-objectifs* ou optimisation de *Pareto* peut être définie mathématiquement de la façon suivante : à chaque solution X_i on associe le vecteur d'évaluation F défini par :

$$F = (F_1(X_i), ..., F_N(X_i))$$

avec F_k le $k^{ème}$ objectif. La variable N est le nombre d'objectifs.

Une solution X_1 domine une autre solution X_2 si :

$$\forall j : F_j(X_1) \geq F_j(X_2) \text{ et } \exists k : F_k(X_1) > F_k(X_2)$$

avec $(j, k) \in \{1...N\}^2$.

Aucune solution parmi X_1 et X_2 ne domine l'autre si :

$$\exists m_1, m_2 : F_{m_1}(X_1) > F_{m_1}(X_2), F_{m_2}(X_2) > F_{m_2}(X_1)$$

La *frontière de Pareto* est définie comme l'ensemble des solutions non dominées. Le but d'un algorithme d'optimisation multi-objectifs est d'approximer au mieux cette frontière.

Rang de Pareto. Le rang de Pareto d'une solution S est fonction du nombre de solutions que S domine. Voici comment les solutions sont affectées d'un rang de Pareto : le rang 1 est affecté dans l'ensemble des solutions à toutes les solutions non-dominées. Ensuite, ces solutions sont ignorées et le même processus est répété aux solutions restantes auxquelles le rang 2 est affecté, etc... Le processus s'arrête lorsqu'il ne reste plus de solutions dans l'ensemble des solutions ou bien lorsqu'il ne reste que des solutions non dominées.

Convexité. Un ensemble S est convexe si, étant donné deux points distincts quelconques de cet ensemble, le segment qui relie ces deux points est contenu dans l'ensemble S.

Concavité. Un ensemble S est concave s'il existe deux points distincts de cet ensemble tels que le segment qui les relie n'est pas totalement contenu dans l'ensemble S.

La section suivante présente les différentes méthodes existantes en optimisation multi-objectifs.

5.1.3 Méthodes

Introduction

Lorsque l'on considère un problème multi-objectifs et que l'on souhaite se ramener à un seul objectif comme il est d'usage avec les méthodes mono-objectif, modéliser un problème sous la forme d'une unique équation peut s'avérer être une tâche difficile étant donné que ce domaine s'attache à optimiser une seule et unique facette du problème considéré, ce qui n'est pas forcément le cas lorsque l'on considère des problèmes plus complexes possédant de multiples caractéristiques devant être prises en compte simultanément. Cet aspect constitue la principale difficulté de l'optimisation mono-objectif. L'optimisation multi-objectifs offre des possibilités supplémentaires qui font défaut à l'optimisation mono-objectif. Le revers de la médaille de cette souplesse est qu'au lieu d'obtenir une solution unique, les méthodes d'optimisation multi-objectifs fournissent une multitude de solutions (également qualifiée de *compromis*), chacune d'entre elles étant ni meilleure ni pire qu'une autre. Ces solutions sont dénommées les *solutions de Pareto* et l'ensemble de ces solutions obtenues à l'issue de l'exécution de l'algorithme d'optimisation est un sous-ensemble de la *surface de compromis*.
L'obtention de cet ensemble de solutions engendre un problème. En effet, il est nécessaire de pouvoir faire une sélection parmi les solutions trouvées par la méthode d'optimisation multi-objectifs mise en œuvre. Celle qui est finalement choisie par l'utilisateur reflètera les compromis opérés par le décideur vis-à-vis des différentes fonctions objectif.
Le décideur étant « humain », il va faire des choix et l'un des buts de l'optimisation multi-objectifs va être de modéliser ses choix ou encore ses préférences. Il existe deux grandes approches pour opérer cette modélisation :

- L'approche de l'utilité multi-attribut [Keen93].
- L'approche de l'aide à la décision [Roy93].

La première approche cherche à modéliser les préférences du décideur, et repose sur l'idée qu'implicitement chaque décideur cherche à optimiser une fonction appelée *fonction d'utilité*. A l'issue de cette approche il ne peut y avoir des solutions ex-aequo. La seconde approche cherche quant à elle à reproduire le processus de sélection du/des décideur(s). Pour ce faire, il est nécessaire de mettre en œuvre des outils de sélection de solutions parmi un ensemble de solutions. Cette approche, contrairement à la première, accepte l'obtention de solutions ex-aequo.
Une autre difficulté à laquelle l'utilisateur est confronté avant d'effectuer une optimisation multi-objectifs, est relative au nombre de méthodes existantes. En effet, ces méthodes sont réparties selon trois grandes familles, chacune d'elles étant définie en fonction de l'instant où la décision d'effectuer ou non un compromis entre fonctions d'objectif est prise. Ces trois familles sont les suivantes [Veld99] :

- **Méthodes d'optimisation *a priori***. Dans ces méthodes, le compromis à effectuer entre les différentes fonctions objectif a été déterminé avant l'exécution de la méthode d'optimisation. Pour obtenir la solution du problème, il suffit de réaliser une seule recherche et, en fin d'optimisation, la solution obtenue reflètera le compromis que l'utilisateur désirait effectuer entre les fonctions objectif avant de lancer la recherche. L'avantage de ces méthodes repose sur le fait qu'il suffit d'une seule recherche pour trouver la solution. Cependant, un travail de modélisation du compromis est nécessaire avant d'obtenir la solution. De plus, le décideur étant « humain », ce dernier sera susceptible de constater que la solution obtenue en fin d'optimisation n'est pas satisfaisante, et de fait, souhaitera une autre solution qui reflète un autre compromis entre les fonctions objectif.

- **Méthodes d'optimisation *progressives***. Dans ces méthodes, au cours de l'optimisation, le décideur est questionné afin que celui-ci puisse réorienter la recherche vers des zones susceptibles de contenir des solutions qui satisfont les compromis qu'il souhaite opérer entre les fonctions objectif. Ces méthodes, bien qu'appliquant une technique originale pour modéliser les préférences du décideur, présentent l'inconvénient de monopoliser l'attention du décideur tout au long de l'optimisation. Cet aspect peut représenter un inconvénient sérieux lorsque une évaluation de la fonction d'objectif est importante. En effet, il peut être difficile d'avoir recours à l'attention du décideur pendant des durées relativement longues en lui posant fréquemment des questions.

- **Méthodes d'optimisation *a posteriori***. Dans ces méthodes, un ensemble de solutions bien réparties dans l'espace de recherche est recherché. Le but sera ensuite de proposer ces solutions au décideur afin qu'il puisse faire une sélection de celle qui est la plus satisfaisante parmi les différentes solutions proposées. L'avantage ici est qu'il n'est plus nécessaire de modéliser les préférences du décideur. L'idée est de produire un ensemble de solutions qui sera transmis à celui-ci. Il y a donc un gain de temps sensible vis-à-vis de la phase de modélisation des préférences de la famille des méthodes *a priori*. L'inconvénient est qu'il est nécessaire de générer un ensemble de solutions bien réparties. Cette tâche est difficile mais de plus, peut requérir un temps d'exécution prohibitif.

Description des méthodes

Il existe un nombre important de méthodes que l'on peut classer également de la façon suivante, cette classification étant plus précise que la précédente [Deb01, Coll02] :

- Les méthodes scalaires.
- Les méthodes interactives.
- Les méthodes floues.
- Les méthodes exploitant une métaheuristique.

- Les méthodes d'aide à la décision.

Les méthodes d'optimisation a priori regroupent la plupart des méthodes fusionnant les fonctions objectif en une seule, c'est-à-dire les méthodes scalaires. Les méthodes d'optimisation progressives incluent les méthodes interactives.

Méthodes scalaires. Ces méthodes transforment un problème d'optimisation multi-objectifs en un problème d'optimisation mono-objectif en utilisant des méanismes permettant de fusioner les fonctions objectifs en une seule.

- **Méthode de pondération des fonctions objectif** [Coel98]. Il s'agit de la méthode la « plus évidente » souvent qualifiée de « méthode naïve de l'optimisation multi-objectifs ». Elle consiste à revenir à un problème d'optimisation mono-objectif, pour lequel de nombreuses méthodes éprouvées sont disponibles en appliquant un coefficient à chacune des fonctions objectif et en faisant la somme de ces entités. Une nouvelle fonction objectif est alors optimisée via une méthode d'optimisation mono-objectif.

- **Méthode de Keeney-Raiffa** [Keen93]. Cette méthode utilise le produit des fonctions objectif pour se ramener à un problème d'optimisation mono-objectif et c'est en ce sens qu'elle diffère de la méthode précédente.

- **Méthode de la distance à un objectif de référence** [Azar96]. Cette méthode permet de transformer un problème d'optimisation multi-objectifs en un problème d'optimisation mono-objectif par le biais d'une somme de la forme $\sqrt[k]{\sum ()^k}$. De plus, dans certains cas, les éléments sont normalisés avant d'en faire la somme [Miet99].

- **Méthode du compromis** [Miet99]. Les méthodes précédentes fusionnent les fonctions objectif en une seule (le terme d'agrégation des fonctions objectif est également utilisé). Les méthodes qui suivent font de même mais comportent un certain nombre de contraintes supplémentaires. La présente méthode également dénommée méthode de la contrainte-ϵ, consiste à :

  1. Choisir un objectif à optimiser prioritairement.
  2. Choisir un vecteur de contraintes initial.
  3. Transformer le problème en conservant l'objectif prioritaire et en transformant les autres objectifs en contraintes d'égalité.

- **Méthode hybride** [Esch]. Il s'agit d'utiliser plusieurs des méthodes précédemment décrites pour en obtenir de nouvelles. La plus connue des méthodes hybrides

est la méthode de Corley qui utilise la méthode de pondération des fonctions objectif et la méthode du compromis.

- **Méthode dite « du but à atteindre »** [Pent93]. Cette méthode permet de manipuler des domaines réalisables qui ne sont pas forcément convexes contrairement à la méthode de pondération des fonctions objectif. Cette méthode consiste à :
  1. Choisir un vecteur de fonctions objectif initital $\vec{F}$.
  2. Choisir une direction de recherche $\vec{w}$ qui consiste en un ensemble de coefficients de pondération des fonctions objectif.
  3. Minimiser un coefficient scalaire λ qui représente l'écart par rapport à l'objectif initital $\vec{F}$ qui a été fixé.

- **Méthode dite « du but programmé »** [Azar96, Coel98, Miet99]. Cette méthode est proche de la méthode du but à atteindre, la principale différence étant qu'après avoir transformé la forme du problème d'optimisation, des contraintes d'inégalités remplacent les contraintes d'égalité. Cette méthode consiste en :
  1. Choisir un vecteur de fonctions objectif initital $\vec{F}$.
  2. Associer à chaque objectif, deux nouvelles variables appelées « déviations » par rapport au vecteur de fonctions objectif initial fixé.
  3. Minimiser l'une des deux variables. Le choix de la variable se faisant en fonction du type de dépassement souhaité (au-dessus ou au-dessous de l'objectif fixé).

- **Ordonnancement lexicographique** [Coel98]. Cette méthode, très intuitive, consiste à considérer les fonctions objectif les unes après les autres et à minimiser (ou à maximiser) un problème d'optimisation mono-objectif, en complétant au fur et à mesure l'ensemble des contraintes.

- **Méthode des contraintes d'égalité propres** [Lin76]. On l'appelle aussi la méthode des égalités strictes, elle est complexe du point de vue mathématique et ne requière pas d'hypothèse de linéarité-convexité.

- **Méthode des contraintes d'inégalité propres** [Lin97]. Cette méthode est fortement inspirée de la précédente. Elle transforme le problème de départ en utilisant des contraintes d'inégalité et non plus des contraintes d'égalité. Les algorithmes de Lin-Tabak [Taba79] et de Lin-Giesy [Gies78] sont basés sur cette méthode.

Méthodes interactives. Elles permettent de chercher une et une seule solution. Elles forment la famille des méthodes progressives et permettent à l'utilisateur de déterminer ses préférences vis-à-vis d'un compromis entre objectifs au cours de l'optimisation. Les méthodes interactives comprennent des méthodes suivantes :

- **Méthode du compromis par substitution** [Haim75]. Elle est basée sur la méthode du compromis, à laquelle s'ajoute un processus interactif pour que la méthode converge vers la solution la plus susceptible de satisfaire l'utilisateur.

- **Méthode de Fandel**. Le but de cette méthode est de guider l'utilisateur dans le choix de ses coefficients de pondération.

- **Méthode STEP** [Esch]. Cette méthode similaire à la précédente, permet de restreindre l'espace de recherche étape par étape grâce aux informations sur la préférence de l'utilisateur.

- **Méthode de Jahn** [Esch]. Il s'agit de choisir un point de départ, puis le problème est résolu pas à pas à travers l'espace de recherche en direction de la solution optimale au sens de Pareto. Cette méthode est fondée sur une méthode de minimisation cyclique (aussi appelée « méthode d'optimisation scalaire dans les directions réalisables » de Zoutendjik).

- **Méthode de Geoffrion** [Esch]. Cette approche est similaire à la méthode de Jahn. Elle repose sur l'algorithme de Franck-Wolfe.

- **Méthode du Simplex** [Neld65]. Il s'agit d'une méthode de recherche séquentielle de l'optimum d'un problème d'optimisation. La méthode utilise $k+1$ essais (où k représente la dimension de la variable de décision $\vec{x}$) pour définir une direction d'amélioration des fonctions objectif.

Méthodes floues. Ces méthodes sont basées sur la logique (dite logique floue) élaborée par l'automaticien L. A. Zadeh. Les méthodes floues comprennent les méthodes suivantes :

- **Méthode de Sakawa** [Saka88, Saka02]. Cette méthode fait intervenir la logique floue à tous les niveaux (sur les paramètres du problème ainsi que sur les paramètres des contraintes). L'ensemble de solutions trouvé utilise également la logique floue dans le sens où les solutions auront un niveau d'appartenance, c'est-à-dire une *corrélation* variable avec l'objectif initial (le niveau de corrélation avec l'objectif initial étant fixé par l'utilisateur).

- **Méthode de Reardon** [Rear97a, Rear97b]. Il s'agit d'une méthode simplifiée de traitement multi-objectifs utilisant la logique floue dans laquelle pour chaque fonction objectif, une fonction d'appartenance est définie afin « d'informer » l'algorithme que la fonction objectif a sa valeur située dans un intervalle centré sur une valeur particulière appelée valeur expérimentale, il s'agit de la valeur qui doit être égale à la valeur de la fonction.

Métaheuristiques. Celles-ci sont des méthodes de recherche dédiées en général aux problèmes d'« optimisation difficile ». Ces méthodes sont, le plus souvent, présentées sous la forme de concepts. Elles reprennent certaines idées présentes dans la vie courante ou dans la nature. Les méthodes basée sur la mise en œuvre d'une métaheuristique sont les suivantes :

- **Recuit simulé** [Bono88, Siar94, Siar95]. Cette méthode proposée par des chercheurs d'IBM qui étudiaient les verres de spin, utilise un processus métallurgique (le recuit) pour trouver un minimum. Ce processus a été transposé à l'optimisation et a donné une méthode simple et efficace.

- **Recherche Tabou** [Glov86, Glov90, Glov97]. Cette méthode mise au point par F. Glover, est conçue en vue de surmonter les minima locaux de la fonction objectif. C'est une technique d'optimisation combinatoire que certains présentent comme une alternative au recuit simulé. Elle consiste, à partir d'une configuration initiale quelconque, à engendrer une succession de configurations qui doit aboutir à la configuration optimale.

- **Algorithmes génétiques** [Holl75, Gold89]. La dernière famille de méthodes exploitant une métaheuristique est celle des algorithmes génétiques que nous avons décrits au chapitre 4.

Toutes les méthodes présentées jusqu'à présent sont basées sur la relation de dominance. Une autre manière de procéder consiste à considérer une relation d'ordre entre les différents éléments.

Les méthodes d'aide à la décision. La première catégorie est composée des méthodes ELECTRE I, IS, II, III, IV et TRI [Vinc89, Mays94] et PROMETHEE I et II [Bran86]. Chacune des méthodes de ces deux familles traite une problématique particulière. Leur différenciation repose d'une part sur la définition de la relation de préférence utilisée et d'autre part sur la méthode d'exploitation des résultats.

Le tableau 5.1 résume toutes les méthodes d'optimisation multi-objectifs présentées. Comme mentionné précédemment, notre choix s'est porté, dans notre approche, sur

Méthodes scalaires	Méthode de pondération des fonctions objectif
	Méthode de Keeny-Raiffa
	Méthode de la distance à un objectif de référence
	Méthode du compromis
	Méthode hybride
	Méthode du but à atteindre
	Méthode du but programmé
	Ordonnancement lexicographique
	Méthode de contraintes d'égalité propres
	Méthode de contraintes d'inégalité propres - Algorithme de Lin-Tabak - Algorithme de Lin-Giesy
Méthodes interactives	Méthode du compromis par substitution
	Méthode de Fandel
	Méthode de STEP
	Méthode de Jahn
	Méthode de Geoffrion
	Méthode du Simplex
Méthodes floues	Méthode de Sakawa
	Méthode de Reardon
Méthodes basées sur une métaheuristique	Recuit simulé
	Recherche Tabou
	Algorithmes génétiques
Méthodes d'aide à la décision	Méthodes ELECTRE
	Méthodes PROMETHEE

TAB. 5.1 – *Les différentes méthodes d'optimisation multi-objectifs.*

les méthodes exploitant une métaheuristique et en particulier sur celles basées sur les algorithmes génétiques. La section suivante présente en détail ces méthodes.

5.2 Optimisation multi-objectifs et algorithmes génétiques

Les algorithmes génétiques sont très bien adaptés au traitement d'un problème d'optimisation multi-objectifs pour plusieurs raisons. En premier lieu, ils manipulent à chaque itération un ensemble de solutions potentielles sur lesquelles ils sont à même d'évaluer les différents objectifs du problème. La mise en œuvre d'une relation comme la relation de dominance permet ensuite de distinguer les solutions non-dominées. Le domaine de l'optimisation multi-objectifs est très dynamique et ne cesse de se déve-

lopper [Fons93, Srin93, Coel98, Deb99]
Dans le chapitre 4 est présenté le schéma de fonctionnement de l'algorithme génétique « standard » à la figure 4.1. Le schéma de l'agorithme génétique « standard » pour la résolution d'un problème d'optimisation multi-objectifs est représenté sur la figure 5.1.

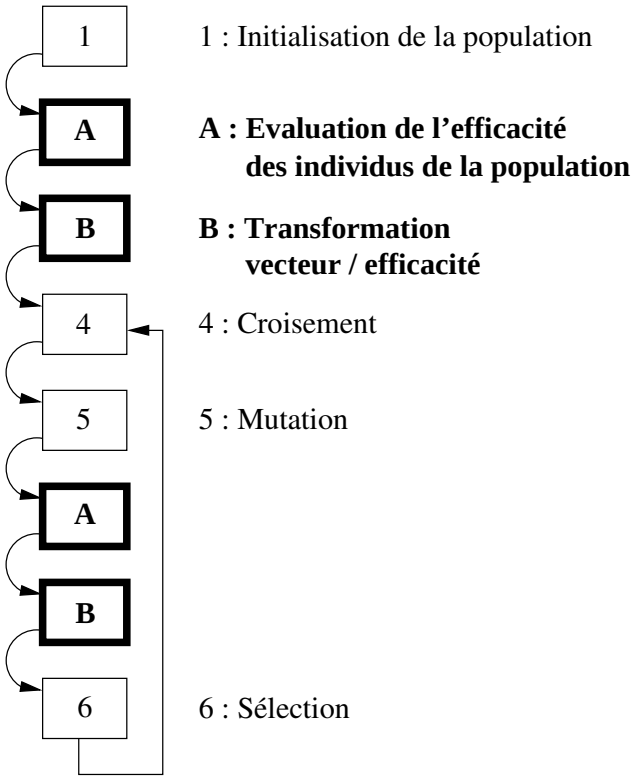


FIG. 5.1 – *Le fonctionnement d'un algorithme génétique multi-objectifs.*

A l'étape B, le vecteur contenant les valeurs des fonctions objectif pour chaque individu est transformé en une efficacité. Diverses stratégies, qui sont décrites dans les sections suivantes, peuvent être adoptées. Les algorithmes génétiques traitant un problème d'optimisation multi-objectifs peuvent en général se ramener à ce schéma.
Il existe deux familles de méthodes pour le traitement de problème d'optimisation multi-objectifs :

- La première famille est composée des méthodes dites « agrégatives » c'est-à-dire celles se ramenant à un problème d'optimisation mono-objectif en utilisant une fonction objectif équivalente.

- La seconde famille comporte les méthodes dites « non agrégatives ». Dans ces méthodes, il n'y a pas fusion des différentes fonctions objectif pour se ramener à

un problème d'optimisation mono-objectif.

Note : la plupart des algorithmes utilisent une « distance » qui caractérise la différence entre deux individus. En effet, les relations utilisées permettant de distinguer les individus non-dominées sont basées sur de telles distances. La distance entre l'individu i et l'individu j est notée $d(i,j)$. On calcule cette distance de la manière suivante : chaque différence entre les motifs (gènes) des individus compte pour $+1$: il s'agit de la distance de Hamming.

Les sections suivantes présentent trois des algorithmes génétiques multi-objectifs existants : VEGA, MOGA et NSGA. Nous présentons uniquement ces trois algorithmes car celui que nous utilisons est basé sur l'algorithme NSGA qui est lui-même basé sur l'algorithme MOGA, ce dernier étant une extension de l'algorithme VEGA.

5.2.1 Algorithme VEGA

L'algorithme VEGA *(Vector Evaluated Genetic Algorithm)* [Scha85] est un algorithme appartenant aux méthodes « non-agrégatives ».

Présentation

L'algorithme VEGA considère une population de N individus. Ces individus sont répartis en k groupes (k étant le nombre de fonctions objectif du problème considéré) de $\frac{N}{k}$ individus [1]. A chaque groupe, est associée une fonction objectif. Cette fonction objectif permet de déterminer l'efficacité d'un individu au sein du groupe. Ensuite, les individus sont mélangés et les croisements sont opérés en tenant compte de l'efficacité de chaque individu.
La figure 5.2 représente les différentes phases de fonctionnement de la méthode. Sur cette figure, sont représentés les différents types de populations qui sont traités au cours du déroulement de la méthode VEGA (soit un ensemble d'individus, soit des groupes d'individus).

Voici les étapes de l'algorithme VEGA :

Etape 1 : Création des k groupes (sous-populations).
Etape 2 : Calcul des efficacités et mélange des individus.
Etape 3 : Application de l'algorithme génétique classique (croisement, mutation sélection).

Avec cette méthode le risque est d'obtenir, en fin d'optimisation, une population constituée d'individus moyens selon chaque objectif. Une telle population ne permet

[1] Avec N multiple de k.

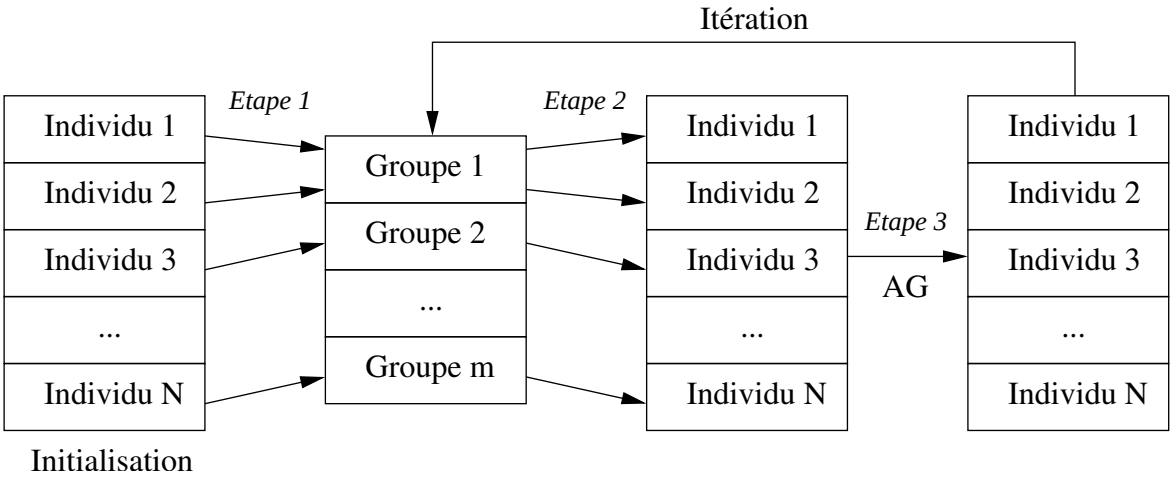


FIG. 5.2 – *Principe de l'algorithme* VEGA.

pas d'obtenir une surface de compromis bien échantillonnée. En effet, la population va se concentrer autour d'un « point » moyen. Des artifices ont été trouvés pour contrebalancer cet effet comme l'utilisation d'espèces au sein du groupe [Deb01].
De plus, il a été montré que cette méthode est équivalente à la méthode de pondération des fonctions objectifs. Donc, elle ne permet pas de trouver des solutions qui se trouveraient dans une concavité [2].

5.2.2 Algorithme MOGA

L'algorithme MOGA *(Multiple Objective Genetic Algorithm)* [Deb99, Fons93], qui est un algorithme « agrégatif », utilise la relation de dominance pour déterminer l'efficacité d'un individu.

Présentation

Cette méthode est basée sur la dominance au sens de Pareto. Ici le « rang » d'un individu (numéro d'ordre qui permet de classer un individu par rapport aux autres) est défini à partir du nombre d'individus qui dominent l'individu considéré dans la population courante.
Par exemple, si l'on considère un individu x à la génération t qui est dominé par P_t individus, le rang de l'individu est défini par :

$$rang(x,t) = 1 + P_t$$

[2] La définition de la concavité est présentée à la section 5.1.2.

A tous les individus non dominés est affecté le rang 1. Les individus dominés se voient donc affectés d'un rang important, c'est-à-dire d'une forte pénalité étant donné que l'objectif est d'approcher la frontière de Pareto (les individus dominés doivent, autant que faire se peut, ne pas être favorisés).

Le calcul de l'efficacité peut suivre les étapes suivantes :

1. Classer les individus en fonction de leur rang.

2. Affecter une efficacité à un individu en interpolant à partir du meilleur (rang 1) jusqu'au plus mauvais (rang n), en utilisant une fonction $f(rang)$ comme illustré sur la figure 5.3). Cette fonction est le plus souvent linéaire.

L'algorithme correspondant est représenté à la figure 5.4.

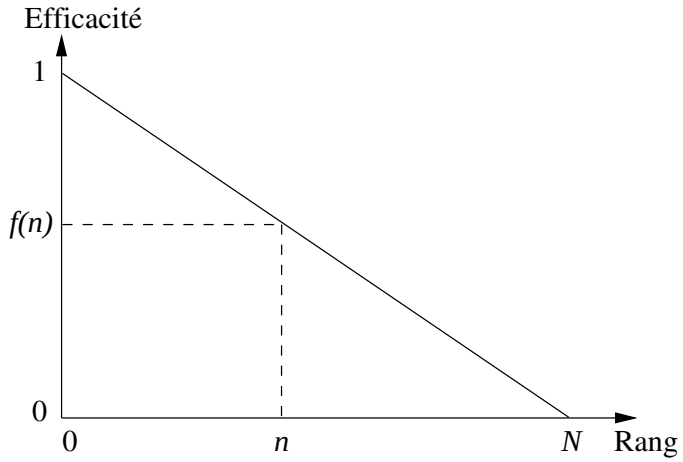


FIG. 5.3 – *Un exemple de fonction d'efficacité.*

La méthode MOGA ne permet pas, dans certains cas, d'obtenir une diversité dans la représentation des solutions car l'algorithme favorise la convergence de la population vers les meilleurs individus. La méthode NSGA, qui va être présentée ci-dessous, permet de contourner cette difficulté.
Pour ce type de méthode, le nombre de comparaisons effectuées par génération est :

$$N \times (N-1) \times n$$

où N correspond au nombre d'individus dans la population et n au nombre de fonctions objectif.

```
1)  Algorithme MOGA
2)  Entrée :
3)      entier N : nombre maximum de générations
4)  Début
5)      Initialisation de la population
6)      Evaluation des fonctions objectif
7)      Assignation d'un rang basé sur la dominance
8)      Assignation d'une efficacité à partir du rang
9)      Pour i de 1 à N
10)         Sélection aléatoire proportionnelle à l'efficacité
11)         Croisement
12)         Mutation
13)         Evaluation des fonctions objectif
14)         Assignation d'un rang basé sur la dominance
15)         Assignation d'une efficacité à partir du rang
16)     Fin pour
17) Fin
```

FIG. 5.4 – *L'algorithme* MOGA.

5.2.3 Algorithme NSGA

L'algorithme NSGA (Non dominated Sorting Genetic Algorithm) [Srin93], qui est un algorithme « agrégatif », reprend les grandes lignes de l'algorithme MOGA précédemment décrit. La différence principale intervient lors du calcul de l'efficacité d'un individu.

Présentation

Cet algorithme est basé sur une classification en plusieurs niveaux des individus. Dans un premier temps, avant de procéder à la sélection, chaque individu de la population est affecté d'un rang (en utilisant le rang de Pareto présenté dans la section 5.1.2). Tous les individus non dominés de même rang sont classés dans un groupe. A ce groupe, est affecté une efficacité factice, qui est inversement proportionnelle au rang de Pareto du groupe considéré.
Les individus d'un groupe doivent se répartir de manière uniforme au sein de celui-ci, autrement dit, il doit y avoir une diversité des solutions.
Pour maintenir la diversité de la population, ces individus classés se voient affectés d'une nouvelle valeur d'efficacité f_i définie par :

$$f_i = \frac{F}{m_i}$$

où F est la valeur d'efficacité affectée au groupe auquel appartient l'individu et :

$$m_i = \sum_{j=1}^{K} P(d(i,j)$$

avec :

$$P(d(i,j)) = \begin{cases} 1 - \left(\frac{d(i,j)}{\delta}\right)^2 & \text{si } d(i,j) < \delta \\ 0 & \text{sinon.} \end{cases}$$

Le paramètre K désigne le nombre d'individus dans la catégorie considérée et $d(i,j)$ est la distance de Hamming entre l'individu i et l'individu j. Il est toujours possible d'utiliser la distance euclidienne à la place de cette dernière. Le paramètre δ est la distance d'influence qui permet de définir des niches écologiques : tous les individus qui sont suffisamment proches, c'est-à-dire dont la distance $d(i,j)$ est inférieure à δ, appartiennent au groupe de l'individu considéré. Les autres seront ignorés.

La figure 5.5 montre les différents paramètres qui interviennent dans le calcul de l'efficacité d'un individu. La zone d'influence d'un individu représente la portion de l'espace contenant les individus à une distance inférieure ou égale à la distance d'influence. Cette zone s'appelle une niche écologique.
Ensuite, les individus de ce groupe sont ignorés. Le processus reprend alors avec les individus restants, où une nouvelle classification est opérée, ainsi qu'une nouvelle catégorisation et une nouvelle opération de partage. Ce processus est répété jusqu'à ce que tous les individus aient été traités.

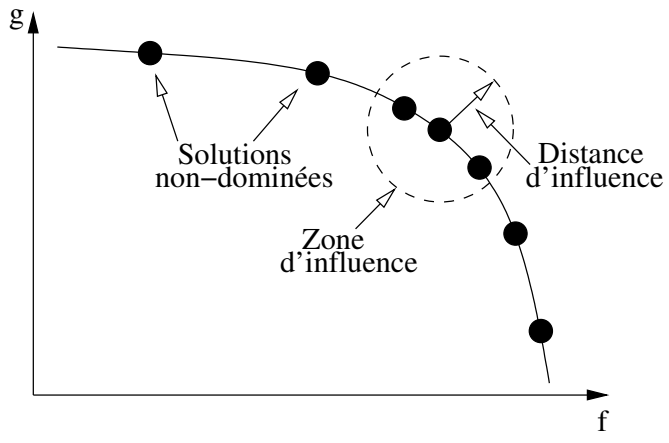


FIG. 5.5 – *Le calcul de l'efficacité.*

Comme les individus qui ont un rang de Pareto de valeur 1 on une meilleure efficacité, ils sont reproduits en plus grand nombre que les autres. Cette propriété permet

d'obtenir une convergence plus rapide vers la surface de compromis. Le partage, lui, permet de maintenir une répartition uniforme sur cette surface. Le pseudo-code de cette méthode est donné dans la figure 5.6.

```
1)  Algorithme NSGA
2)  Entrée :
3)      entier N : nombre maximum de générations
4)  Début
5)      Initialisation de la population
6)      Evaluation des fonctions objectif
7)      Assignation d'un rang basé sur le rang de dominance
8)          sur chaque zone d'influence
9)      Calcul du compte des voisins
10)     Assignation d'une efficacité partagée
11)     Pour i de 1 à N
12)         Sélection aléatoire proportionnelle à l'efficacité
13)         Croisement
14)         Mutation
15)         Evaluation des fonctions objectif
16)         Assignation d'un rang basée sur le rang de dominance
17)             sur chaque zone d'influence
18)         Calcul du compte des voisins
19)         Assignation d'une efficacité partagée
20)     Fin pour
21) Fin
```

FIG. 5.6 – *L'algorithme* NSGA

On distingue trois types de niches :

- Les niches génotypiques :

$$P_{geno}(d(i,j)) = \begin{cases} 1 - \left(\frac{d_{ham}(i,j)}{\delta_{geno}}\right)^2 & \text{si } d_{ham}(i,j) < \delta_{geno} \\ 0 & \text{sinon} \end{cases}$$

  le comptage de niches (de rayon δ_{geno}) est effectué dans l'espace des gènes, la distance considérée est la distance de Hamming ($d_{ham}(i,j)$).

- Les niches phénotypiques :

$$P_{pheno}(d(i,j)) = \begin{cases} 1 - \left(\frac{d_{euclid}(i,j)}{\delta_{pheno}}\right)^2 & \text{si } d_{euclid}(i,j) < \delta_{pheno} \\ 0 & \text{sinon} \end{cases}$$

le comptage de niches (de rayon δ_{pheno}) est effectué dans l'espace des phénotypes, la distance considérée est la distance euclidienne (si l'on considère un problème d'optimisation continu).

- Les niches combinées sont une association des deux types de niches précédentes :

$$P_{comb}(d(i,j)) = \begin{cases} 1 - \left(\frac{d_{ham}(i,j)}{\delta_{geno}} \times \frac{d_{euclid}(i,j)}{\delta_{pheno}}\right)^2 & \text{si } d_{euclid}(i,j) < \delta_{pheno} \\ & \text{et } d_{ham}(i,j) < \delta_{geno} \\ 1 - \left(\frac{d_{euclid}(i,j)}{\delta_{pheno}}\right)^2 & \text{si } d_{euclid}(i,j) < \delta_{pheno} \\ & \text{et } d_{ham}(i,j) \geq \delta_{geno} \\ 1 - \left(\frac{d_{ham}(i,j)}{\delta_{geno}}\right)^2 & \text{si } d_{euclid}(i,j) \geq \delta_{pheno} \\ & \text{et } d_{ham}(i,j) < \delta_{geno} \\ 0 & \text{sinon} \end{cases}$$

le comptage de niches (de rayon δ_{geno} et δ_{pheno}) est effectué dans l'espace des génotypes et des phénotypes.

Une niche phénotypique permet de « régler » la diversité des solutions dans l'espace des fonctions objectifs alors que la niche génotypique permet de régler la diversité des solutions dans l'espace de décision.

L'efficacité de cette méthode tient au fait que les objectifs sont réduits à une valeur d'efficacité factice obtenue en utilisant le classement en fonction du rang de Pareto. Cette méthode présente l'inconvénient d'être sensible au choix de la valeur du paramètre δ.
Pour cette méthode, le nombre de comparaisons effectuées sur une population pour une génération est le même que pour la méthode MOGA. Cependant, un surplus de calculs dû au partage apparaît. Il a été montré que ce surplus est proportionnel à $N \times (N-1)$.

En fouille de données, peu de travaux font usage de l'optimisation multi-objectifs par le biais d'algorithmes évolutionnaires et d'algorithmes génétiques en particulier. Nous présentons dans la section suivante les plus significatifs d'entres eux.

5.3 Expériences en fouille de données

Nous avons présenté dans la section précédente, certains algorithmes évolutionnaires utilisés en optimisation multi-objectifs. Ces algorithmes peuvent être utilisés dans des domaines variés, cependant, en fouille de données, l'optimisation multi-objectifs par le biais de tels algorithmes est peu mise en œuvre à ce jour. Nous présentons ici quelques expériences relatives à l'emploi de techniques évolutionnaires multi-objectifs en fouille de données : une expérience de marketing d'une part, et deux expériences concernant la sélection d'attributs d'autre part.

5.3.1 Optimisation du mailing

Dans [Bhat99, Bhat00], l'auteur présente un problème de modélisation prédictive : il s'agit d'identifier des clients susceptibles de répondre à des sollicications promotionnelles par mailing. Ce domaine nécessite la prise en compte d'objectifs multiples, comme par exemple la fréquence de réponse du client à des sollicitations antérieures ou encore son aptitude à générer le plus de revenus pour l'organisme. L'auteur propose un algorithme évolutionnaire multi-objectifs pour obtenir un modèle non-dominé. Son approche est motivée par le fait que la plupart des approches standards dans ce domaine ne traitent qu'un seul objectif alors que les problèmes de décision « du monde réel » en comportent plusieurs et que les preneurs de décision recherchent des compromis acceptables parmi ces objectifs.
L'algorithme génétique utilisé met en œuvre la dominance au sens de Pareto pour la sélection des individus. Il utilise d'autre part l'élitisme afin de garantir la survie de la totalité des individus non-dominés.
L'algorithme peut employer deux types de représentation pour les individus. Ainsi, un individu peut être la représentation d'un modèle sous l'une des deux formes suivantes :

- Une combinaison linéaire d'attributs de prédiction.
- Des relations non-linéaires comme celles utilisées en programmation génétique [3].

Les opérateurs de croisement et de mutation utilisés sont classiques pour les deux types de représentation des individus. En ce qui concerne la première représentation, le croisement est un croisement à un site et la mutation ne modifie qu'un gène de l'individu avec une probabilité fixée. Concernant la seconde représentation, les deux opérateurs sont conçus de manière à ne pas engendrer d'individus non valides.
Cette approche est exclusive dans la mesure où l'auteur se place dans un contexte très particulier du marketing.

[3] Dans le chapitre 4 à la section 4.1.1, nous présentons les principes de la programmation génétique.

5.3.2 Sélection d'attributs et courbe ROC

Le problème de la sélection d'attributs en fouille de données est un problème présentant de multiples objectifs devant être optimisés de façon simultanée. On peut citer comme critères, la minimisation de la cardinalité des ensembles d'attributs et la maximisation des performances. L'auteur [Emma01] propose ici une extension de l'algorithme génétique NPGA *(Niched Pareto Genetic Algorithm)* [Horn93] pour traiter le problème de la sélection d'attributs en particulier pour le diagnostic médical et en ingénierie. Les performances d'un classifieur peuvent être évaluées par l'analyse de courbe ROC, en termes de *Sensisiblité* et de *Spécificité* du modèle. Les courbes ROC (Receiver Operating Characteristic) permettent d'étudier les variations de la *Sensisiblité* et de la *Spécificité* d'un test pour différentes valeurs du seuil de discrimination. Le terme de courbe ROC peut être envisagé comme une « courbe de caractéristiques d'efficacité ». L'auteur se place dans le cadre de la modélisation prédictive et fournit une méthode qui repose sur un algorithme évolutionaire basé sur l'algorithme NPGA afin de prendre en compte ces deux critères.
Cet algorithme est non-élitiste, or l'élitisme est un aspect à prendre en compte dans un algorithme génétique multi-objectifs. Ainsi, les solutions non-dominées sont ici conservées dans une archive et participent à la reproduction. Le codage des individus consiste en une combinaison linéaire d'attributs de prédiction. Le croisement est un croisement à deux sites, et la mutation est une mutation en un point.

Cette approche est axée sur les qualités globales du modèle de classifieur et ne prend en compte que les critères de *Sensibilité* et de *Spécificité*.

5.3.3 Sélection d'attributs et arbre de décision

Dans [Papp02] l'auteur propose également un algorithme génétique multi-objectifs basée sur l'algorithme NPGA [Horn93] dédié à la sélection d'attributs. Le but de cet algorithme est de trouver le meilleur sous-ensemble d'attributs qui minimisent à la fois le *Taux d'erreur* et la taille de l'arbre découvert par l'algorithme de classement qui implémente la méthode C4.5 [Quin96], en utilisant la dominance au sens de Pareto.
Dans cet algorithme génétique, chaque individu représente un sous-ensemble d'attributs et est codé sous la forme d'une chaîne de M gènes où M est le nombre d'attributs dans le jeu de données. Chaque gène peut prendre la valeur 1 ou 0 indiquant si l'attribut est présent dans le sous-ensemble représenté par l'individu.

La fonction d'adaptation consiste en deux mesures de qualité : le *Taux d'erreur* du classifieur et la taille de l'arbre de décision construit.

La sélection consiste d'abord en la sélection des individus non-dominés de la population courante. Ces individus sont passés tels quels à la génération suivante par élitisme. Les individus restant sont reproduits par application de la sélection par tournoi à deux individus. La sélection du meilleur individu est basée sur la dominance au sens de

Pareto en prenant en compte les deux objectifs à minimiser. Si un des deux individus domine l'autre, cet individu est bien entendu sélectionné. Si aucun des deux ne domine l'autre, un troisième critère de sélection est utilisé : il s'agit du nombre d'individus dominés par chacun des deux individus qui « s'affrontent » dans le tournoi. L'individu sélectionné étant celui qui domine le plus d'individus. Si les deux individus dominent le même nombre d'individus, un choix aléatoire est effectué afin de les départager.

Le croisement et la mutation sont « classiques » et sont utilisés respectivement avec un taux de 80% et de 1%. Le résultat de l'algorithme consiste en un ensemble d'individus respectant la contrainte suivante : seuls sont gardés les individus non dominés sur l'ensemble de données constitué de l'ensemble d'apprentissage et de l'ensemble de validation.
Notre approche diffère de celle-ci dans le sens où d'une part, nous ne nous intéressons pas à trouver un sous-ensemble d'attributs mais des règles de dépendance et d'autre part, nous recherchons des règles individuellement bonnes sans évaluer le modèle dans sa globalité.

Comme nous l'avons évoqué, les travaux relatifs à la mise en œuvre d'algorithmes évolutionnaires multi-objectifs en fouille de données sont rares. De plus, les travaux existants sont généralement dédiés à un domaine particulier. Notre approche se focalise plutôt sur l'intérêt et la qualité des règles considérées d'un point de vue général sans critères subjectifs liés au domaine d'application. La section suivante présente notre approche basée sur l'algorithme NSGA ainsi que les résultats obtenus.

5.4 Algorithmes génétiques multi-objectifs pour l'extraction de règles

Cette section est relative à l'application de l'algorithme génétique que nous avons implémenté dans un contexte multi-objectifs sur divers jeux de données. Nous nous sommes intéressés à la recherche de règles de dépendance devant satisfaire des critères c'est-à-dire des objectifs, qui sont dans certains cas contradictoires.
Nous présentons dans un premier temps l'algorithme génétique. Nous distinguons le cas des règles *simples* et celui des règles *généralisées* présentées au chapitre 3. Les résultats présentés sont issus d'expériences effectuées sur les trois jeux de données précédents ; l'approche expérimentale, délibéremment choisie, nous a permis de mettre en évidence des phénomènes qui ne se reproduisent pas d'un jeu de données à l'autre et qui sont donc plus difficilement décelables par une approche analytique.

5.4.1 Algorithme

L'algorithme que nous avons implémenté en langage C++ à partir de la bibliothèque EO Evolutionary Computation Framework [eo] est basé sur l'algorithme NSGA

(Non dominated Sorting Genetic Algorithm) [Srin93] présenté en détails à la section 5.2.3. Le codage des individus ainsi que les opérateurs de croisement sont identiques à ceux de l'algorithme génétique présenté dans le chapitre précédent. Nous avons ajouté à l'algorithme génétique un mécanisme d'archivage des solutions non-dominées. En effet, afin de ne perdre aucune des solutions non-dominées trouvées par l'algorithme génétique au cours de son exécution, il est nécessaire de les stocker dans une archive. En ce qui concerne les opérateurs de croisement et de mutation, ceux-ci doivent être adaptées aux deux représentations de règles que nous utilisons (règles *simples* et *généralisées*). La sélection est identique à celle de l'algorithme NSGA. Celle-ci consiste en une sélection aléatoire proportionnelle au rang de Pareto.

Archivage des solutions non-dominées

Le but de l'algorithme génétique est bien entendu de déterminer la meilleure approximation de l'ensemble des solutions de Pareto. Idéalement, cet ensemble doit être contenu dans la population finale de l'algorithme génétique à l'issue de son exécution. Néanmoins, ce résultat n'est pas garanti. En effet, la nature stochastique de l'algorithme peut provoquer la disparition via l'opérateur de mutation, d'individus non dominés. D'autre part, l'étendue de la frontière de Pareto n'est pas nécessairement connue a priori et, de fait, le dimensionnement de la population ne peut être réalisé précisément. Ces deux phénomènes nous ont conduit, en nous inspirant de la technique utilisée dans le recuit simulé [Bono88, Siar94, Siar95], à utiliser un mécanisme d'archivage des solutions non dominées trouvées durant l'exécution de l'algorithme génétique.
Lorsqu'une telle solution est rencontrée suite au croisement ou à la mutation, elle est stockée dans l'archive, une mise à jour de celle-ci étant éventuellement effectuée. Ainsi, les solutions présentées à l'utilisateur sont celles se trouvant dans l'archive et non celles de la population finale.

5.4.2 Expériences

Nous avons appliqué l'algorithme génétique aux trois jeux de données présentés précédemment « Vote », « Abonnés » et « Sonar ». Ces jeux de données, aux caractéristiques très différentes, vont nous servir à illustrer les résultats expérimentaux que nous avons également obtenus sur d'autres jeux de données. Dans le chapitre 3, un certain nombre de mesures ont été présentées ainsi que leurs comportements relatifs visualisés sous la forme de graphes bi-dimensionnels obtenus par la méthode du tirage aléatoire. L'objectif du présent chapitre est de se concentrer sur l'extraction des règles optimisant plusieurs mesures qui présentent des antagonismes, afin de découvrir celles qui sont les meilleures. L'analyse de nos résultats est à la fois globale, dans le sens où elle est principalement basée sur l'analyse de graphes, et locale, par l'analyse des règles obtenues. En effet, certaines exécutions peuvent fournir de grandes quantités de règles mais le cas échéant, leur analyse individuelle permet de fournir des renseignements intéressants notamment par l'étude de leur forme. L'étude sémantique des

règles permettant d'obtenir des tendances intéressantes pour l'utilisateur, dépasse le cadre de cette thèse qui se focalise sur les critères objectifs de sélection. Ce sujet constitue d'ailleurs une des perspectives de travaux futurs.

Voici le paramètrage de l'algorithme génétique commun pour les trois jeux de données :

- Distance d'influence : 0,0125.
- Taux de croisement : 0,8.

Le taux de mutation pour un attribut est 1/17 pour le jeu de données « Vote », 1/19 pour le jeu de données « Abonnés » et 1/61 pour le jeu de données « Sonar ». Les valeurs 17, 19 et 61 étant respectivement le nombre d'attributs de chaque jeu de données c'est-à-dire également les longueurs de génotype des individus pour chaque jeu. Il est d'usage de choisir l'inverse de la longueur du génotype comme taux de mutation. Les graphes présentent les points obtenus par l'algorithme d'induction d'arbres de décision, celui-ci ayant été paramétré de façon à fournir un arbre élagué.
Les deux sections suivantes présentent les résultats que nous avons obtenus par application de l'algorithme multi-objectifs optimisant deux mesures sur les trois jeux de données. Cette analyse des résultats se scinde en deux parties : une dédiée à l'étude des graphes et l'autre plus spécifique à des exemples de règles.

5.4.3 Résultats : graphes

Les mesures que nous avons introduites dans le chapitre 3 ont été choisies parce qu'elles présentent de par leurs espaces de recherche des particularités justifiant l'emploi de l'approche multi-objectifs. Les espaces de recherche représentés graphiquement sont constitués de règles générées par le biais de la méthode du tirage aléatoire. Le tableau 3.1 fournit les caractéristiques des attributs des jeux de données dont dépend la taille de ces espaces de recherche. Ces règles sont utiles pour l'estimation de la densité de l'espace de recherche selon les valeurs des mesures bien qu'elles ne rendent pas, une vision exhaustive de celui-ci. Elles permettent de mieux appréhender le comportement des algorithmes de recherche de règles d'association (avec la $Confiance$, le $Lift$ ou encore la *Conviction* comme mesures de sélection) ou d'algorithmes d'induction d'arbres de décision avec ou sans élagage.
Nous avons établi expérimentalement pour chaque jeu de données et pour chaque couple de mesures utilisés, un paramétrage « type » de l'algorithme génétique pour lequel les frontières de Pareto obtenues sont les meilleures du point de vue de la diversité des points qu'elles contiennent mais également du point de vue de leur étendue. Les tableaux 5.2 et 5.3 résument les principales caractéristiques des nuages de points concernant respectivement les règles simples et les règles généralisées et ce pour chaque jeu de données. Chaque ligne comporte les informations pour un couple de mesures. Le nombre de points figurant sur la frontière de Pareto ainsi que la taille de la population et le nombre de générations de l'algorithme génétique sont indiqués. Enfin, la référence

vers la figure comportant les graphes est précisée. Il ressort de ce tableau que les frontières de Pareto associées aux couples de mesures sur les jeux de données « Abonnés » et « Sonar » sont composées de beaucoup plus de solutions que celles de « Vote » en raison de la nature des attributs qui composent ces jeux.

Couple de mesures	*Nombre de points sur la frontière*			*Figures*
	Vote	*Abo.*	*Son.*	
$(Sensibilité, Spécificité)$	12	1625	167	5.7
$(Conviction, Support)$	10	184	46	5.8
$(Lift, Support)$	12	130	149	5.9
$(Sebag, RI)$	9	132	28	5.10
$(Sensibilité, Lift)$	12	189	115	5.11
$(Sensibilité, Sebag)$	10	91	88	5.12

TAB. 5.2 – *Règles simples : Caractéristiques des graphes obtenus sur les trois jeux de données. La taille de la population est de 200, 300 et 350 individus respectivement pour les jeux de données « Vote », « Abonnés » et « Sonar ». De même les nombre de générations sont respectivement de 100, 400 et 400 pour ces trois jeux.*

L'ensemble des points (règles) trouvés par l'algorithme génétique sont pour tous les couples de mesures, au delà du nuage de points fourni par la méthode du tirage aléatoire ou à la limite de ce nuage. Nous avons effectué des exécutions de l'algorithme génétique dix fois plus « poussées » en termes de dimension *(2000 individus)* de la population initiale ainsi que du point de vue du nombre d'exécutions *(1000 générations)* pour certains couples, notamment $(Sensibilité, Spécificité)$ sur le jeu de données « Vote », et le résultat obtenu n'a pas apporté de progrès par rapport à celui dont les paramètres sont précisés dans le tableau 5.2.

Concernant le couple $(Sensibilité, Spécificité)$ à la figure 5.7 : les points de la frontière de Pareto du graphe (a) obtenu sur le jeu de données « Vote » sont de bons compromis compte tenu des valeurs de ces points pour les deux mesures. En effet, sur cette frontière, tous les points ont une *Sensibilité* supérieure à 0.7 et une *Spécificité* supérieure à 0.6. En revanche, les points de la frontière du graphe (b) associé au jeu de données « Abonnés » sont très diversifiés. Ce phénomène est commun à plusieurs couples de mesures sur ces deux jeux de données ; de fait, l'ensemble des règles constituant la frontière de Pareto de chacun de ces graphes, doit subir un post-traitement consistant à rejeter certaines règles qui sont de toute évidence « mauvaises » pour l'utilisateur bien qu'étant des compromis. En effet, les règles aux « extrémités » de la frontière de Pareto ne sont optimisées que pour une seule mesure, ce que l'utilisateur ne recherche pas nécessairement. Ainsi, il est préférable dans de telles configurations

Couple de mesures	*Nombre de points sur la frontière*			*Nombre d'individus*			*Nombre de générations*			*Figures*
	Vote	*Abo.*	*Son.*	*Vote*	*Abo.*	*Son.*	*Vote*	*Abo.*	*Son.*	
(*Conviction*, *Support*)	6	39	-	200	300	-	100	400	-	5.13
(*Sebag*, *Support*)	4	-	9	200	-	300	100	-	400	5.14
(*Sebag*, *RI*)	6	155	30	200	300	300	100	400	400	5.15
(*Sensibilité*, *Sebag*)	4	-	3	200	-	300	100	-	400	5.16

TAB. 5.3 – *Règles généralisées : Caractértistiques des graphes obtenus sur les trois jeux de données. Un tiret signifie qu'il n'y a pas d'antagonisme pour le couple de mesures considéré alors que concernant les règles simples (cf. tableau 5.2) il y a antagonisme dans tous les cas.*

de la frontière de Pareto, de ne choisir les solutions que parmi les règles constituant la partie « centrale » de la frontière.
Les grandes variations quant à la quantité de points sur les frontières de Pareto d'un jeu de données à l'autre sont liées à la dimension des espaces de recherche. En effet, comme on peut le voir dans les tableaux 5.2 et 5.3, les frontières de Pareto associées au jeu de données « Abonnés » comportent beaucoup plus de points que pour les autres jeux en raison de la taille de son espace de recherche. Les jeux de données « Abonnés » et « Sonar » sont constitués en majorité d'attributs réels contrairement à « Vote » principalement constitué d'attributs binaires. Le graphe (c) de la figure 5.7 met en évidence la faiblesse de l'approximation de l'espace de recherche et la force de l'algorithme génétique. En effet, la frontière de Pareto est constituée de points relativement éloignés du nuage.

Concernant le couple (*Conviction*, *Support*) à la figure 5.8 : les points des frontières de Pareto des graphes associés aux jeux de données « Vote » et « Abonnés » sont largement étendus. Notons toutefois que dans le cas du graphe de « Vote », l'utilisation d'un seuil ne concernerait que la *Conviction*, les valeurs de *Support* étant quant à elles sensiblement les mêmes à 0.1 près.

Concernant le couple ($Lift$, *Support*) à la figure 5.9 : les points de la frontière de Pareto sont de bonnes solutions pour les deux mesures sur le graphe (a) du jeu de données « Vote ». Les valeurs de $Lift$ sont comprises entre 2 et 2.5 pour la majorité des points de la frontière, 2.5 étant la valeur maximale pour cette mesure sur ce jeu. De même la majorité des points a un *Support* compris entre 0.25 et 0.4, pour une valeur maximale de celui-ci égale à 0.4. Concernant les graphes (b) et (c) des jeux « Abonnés » et « Sonar », les points sont très diversifiés ce qui nécessite un post-traitement.

Concernant le couple (*Sebag*, *RI*) à la figure 5.10 : sur le graphe de « Vote », la

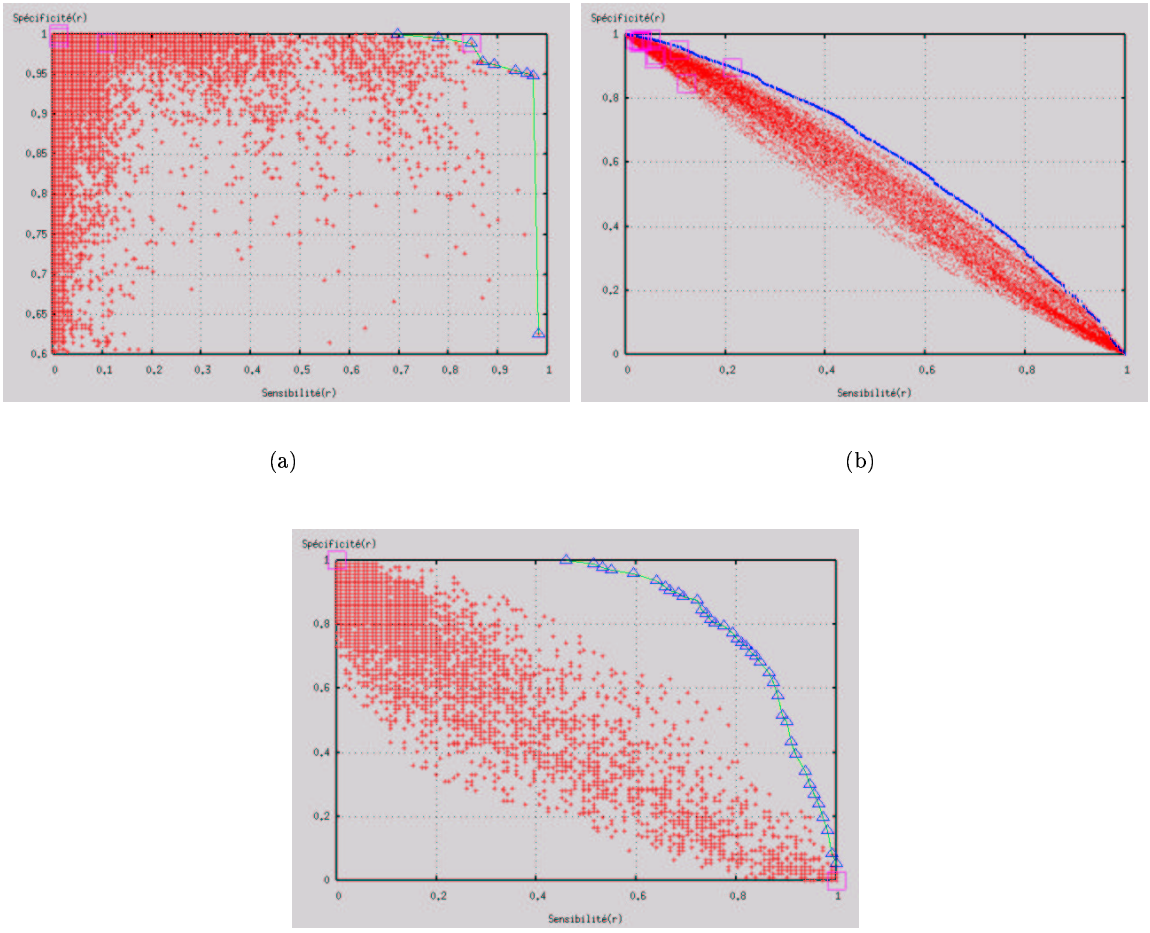
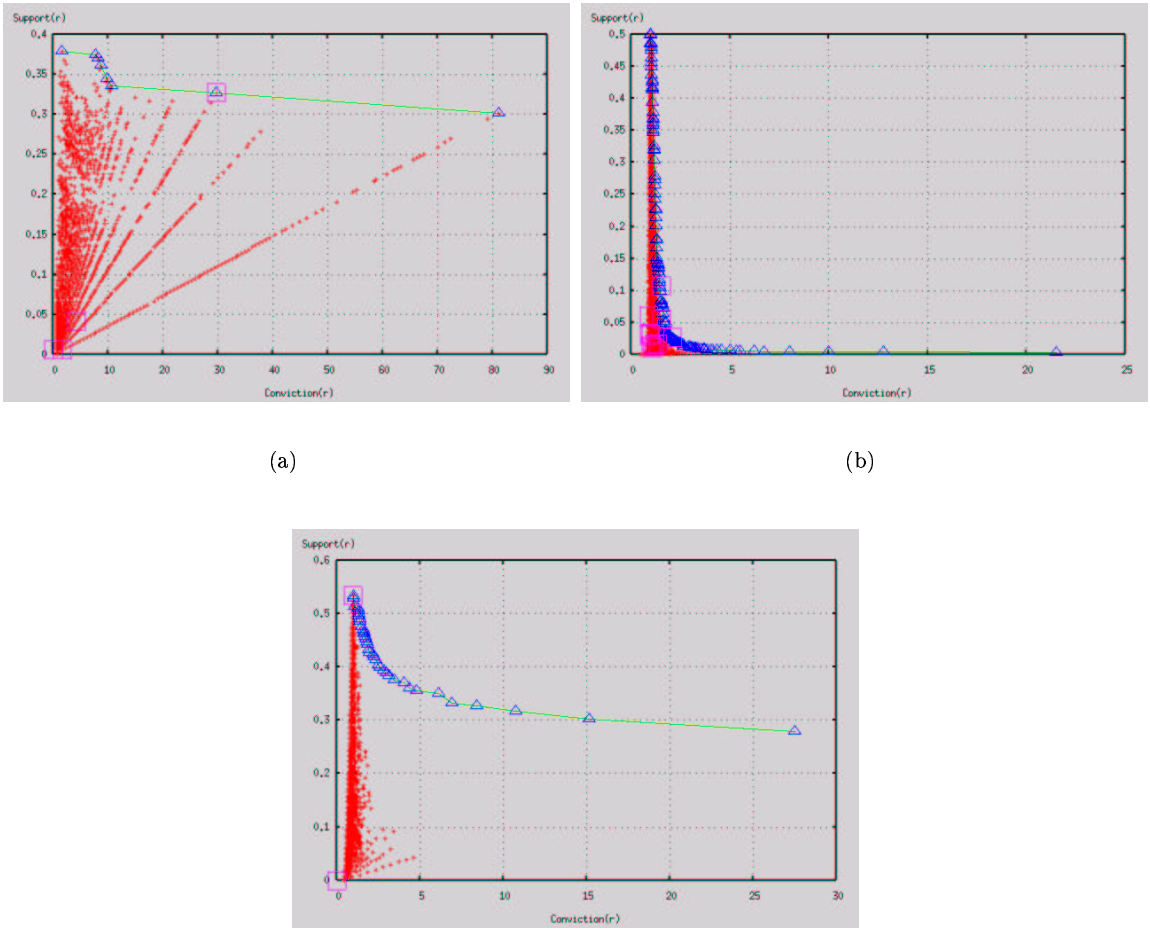


FIG. 5.7 – *Règles simples : Sensibilité versus Spécificité sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs. Carrés : algorithme d'induction d'arbres de décision.*

frontière de Pareto est étendue mais constituée de peu de points. Ceux-ci sont groupés, exepté pour une valeur extrême du coefficient de *Sebag*. Sur le graphe de « Abonnés », la frontière est très diversifiée. Les points de la frontière sont très nombreux sur le graphe (b) du jeu « Abonnés ». Sur le graphe (c) du jeu de données « Sonar », les points aux extrémités de la frontière de Pareto sont acceptables a priori.

Concernant le couple $(Sensibilité, Lift)$ à la figure 5.11 : les points sur la frontière de Pareto sont de bons compromis en opposition à ceux de la frontière des graphes (b) et (c) des jeux « Abonnés » et « Sonar » qui offrent de grandes amplitudes de valeurs.



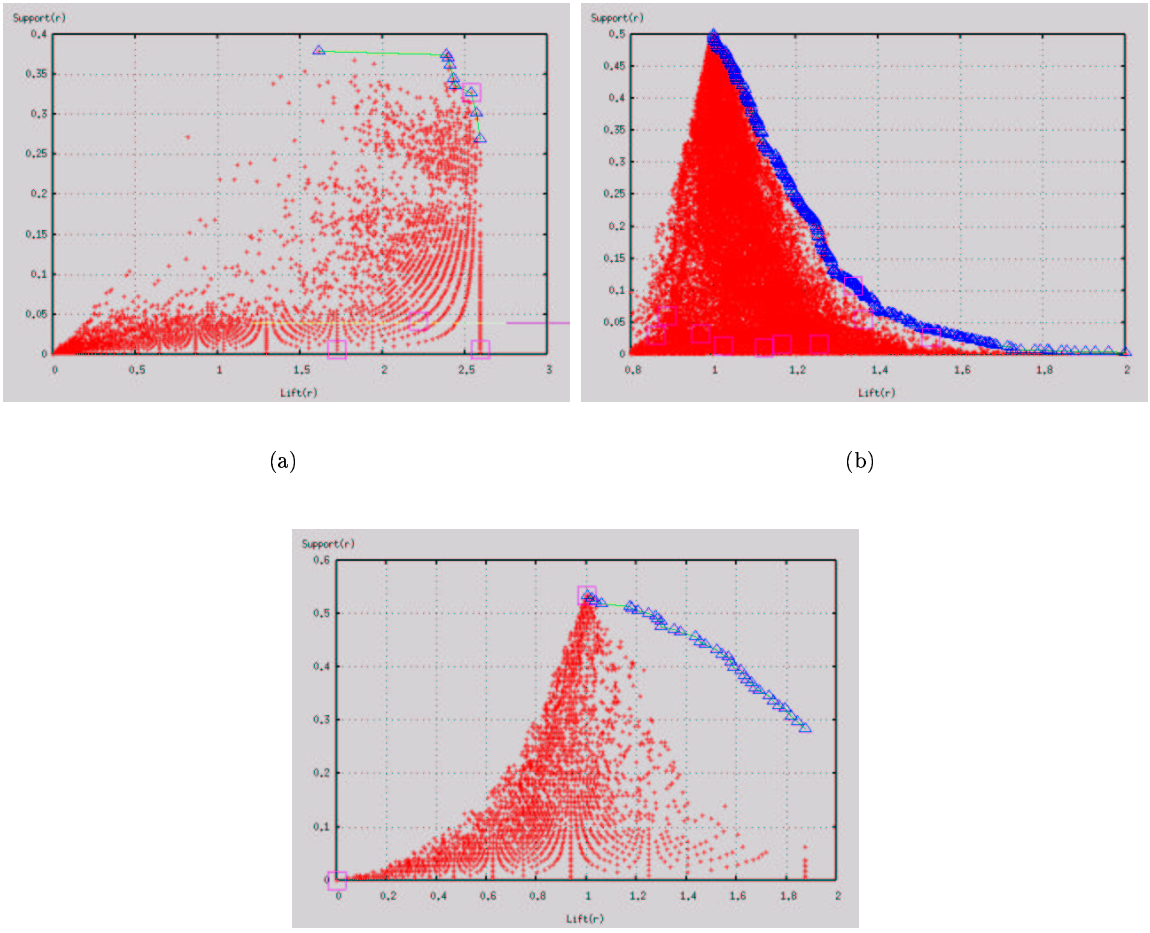
(a) (b)

(c)

FIG. 5.8 – *Règles simples : Conviction versus Support sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs. Carrés : algorithme d'induction d'arbres de décision.*

De même, le couple $(Sensibilité, Sebag)$ à la figure 5.12 présente une frontière de de Pareto dont les points sont de bons compromis. Par contre, ceux des frontières des graphes (b) et (c) de « Abonnés » et « Sonar » offrent de grandes amplitudes de valeurs également.

Nous présentons ici les résultats obtenus sur les mêmes jeux de données que précédemment mais pour la recherche de règles généralisées. Nous comparons les règles obtenues avec celles fournies par l'algorithme de recherche de règles d'associations. Ce der-

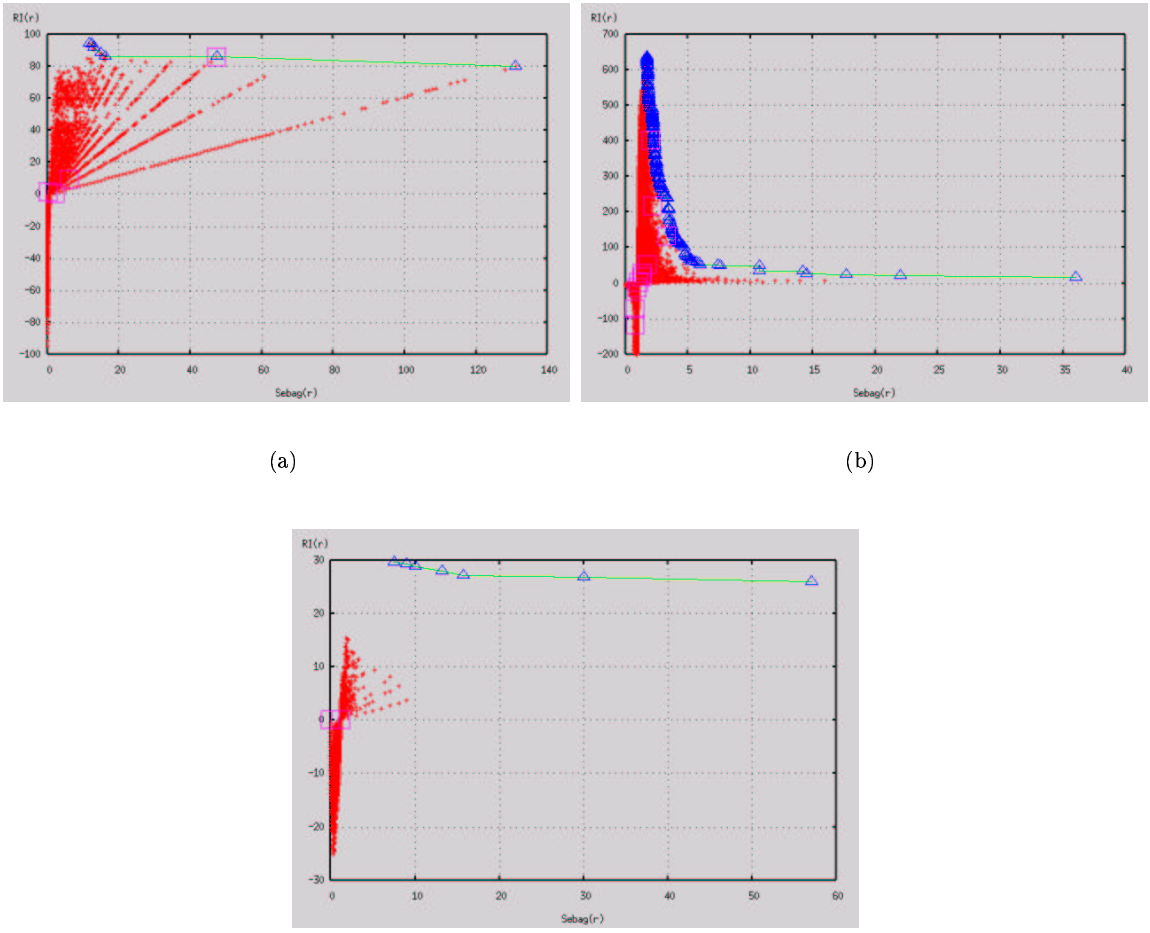


(a) (b)

(c)

FIG. 5.9 – *Règles simples : Lift versus Support sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs. Carrés : algorithme d'induction d'arbres de décision.*

nier a été appliqué avec les trois mesures de : $Confiance$, $Lift$ et $Conviction$. D'autre part, les couples envisagés dans la section précédente ne sont pas tous utilisés ici car ils ne présentent plus d'antagonismes lorsque la taille du conséquent est supérieure à 1. Nous utilisons donc les couples $(Conviction, Support)$, $(Sebag, Support)$, $(Sebag, RI)$ et $(Sensibilité, Sebag)$ pour le jeu de données $Vote$ et les couples $Conviction$ versus $Support$ et $Sebag$ versus RI pour les jeux « Abonnés » et « Sonar ».
Rappelons que l'algorithme génétique applique ici les opérateurs de croisement et de mutation dédiés aux règles généralisées. En effet, pour deux individus, le premier opérateur doit prendre en considération les localisations respectives des gènes correspondants



(a) (b)

(c)

FIG. 5.10 – *Règles simples : Sebag versus RI sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs. Carrés : algorithme d'induction d'arbres de décision.*

à l'antécédent et au conséquent des règles codées afin de ne pas générer des règles sans conséquent ou bien sans antécédent.
L'opérateur de mutation quant à lui, doit éviter de muter un gène d'antécédent en un gène de conséquent, lorsque ce gène est le seul gène antécédent et vice versa.
Comme précédemment, nous présentons ici une description des nuages obtenus.

Concernant le couple $(Conviction, Support)$ à la figure 5.13 : les frontières des graphes (a) et (b) de « Vote » et « Abonnés » sont assez étendues dans les deux cas, mais offrent de faibles diversités de points.

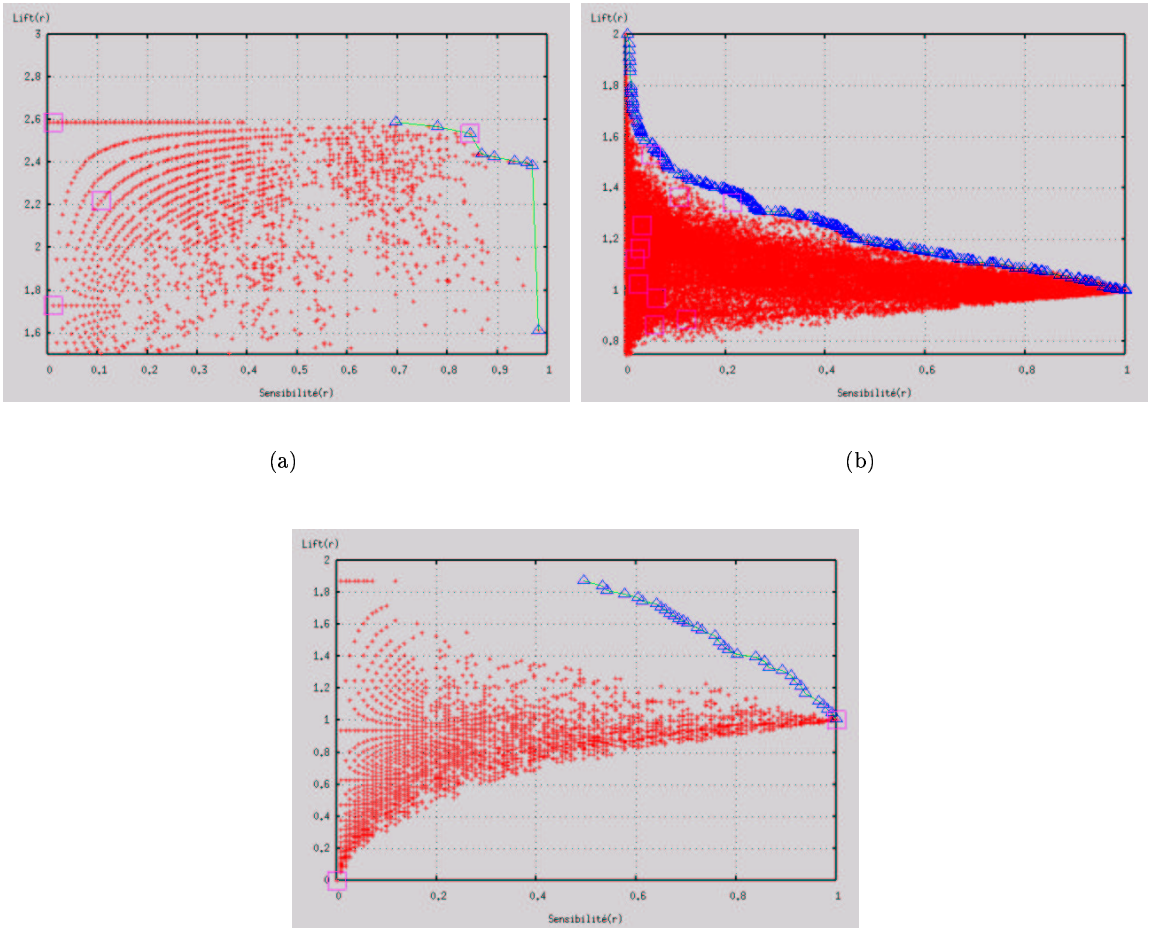


FIG. 5.11 – *Règles simples : Sensibilité versus Lift sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs. Carrés : algorithme d'induction d'arbres de décision.*

Concernant le couple (*Sebag*, *Support*) à la figure 5.14 : seul le graphe sur le jeu de données « Vote » est présenté.

Concernant le couple (*Sebag*, *RI*) à la figure 5.15 : la frontière de Pareto du graphe (a) de « Vote » présente une faible diversité de points en opposition à celle du graphe (b) de « Abonnés », dont le nombre élevé de solutions nécessite un filtrage sur le coefficient *RI*. En effet, de nombreuses règles ont une valeur de *RI* entre 1000 et plus de 2500, pour des valeurs de *Sebag* proches. Ces règles étant des compromis, il est nécessaire de choisir une ou plusieurs d'entre elles selon un autre critère de sélection

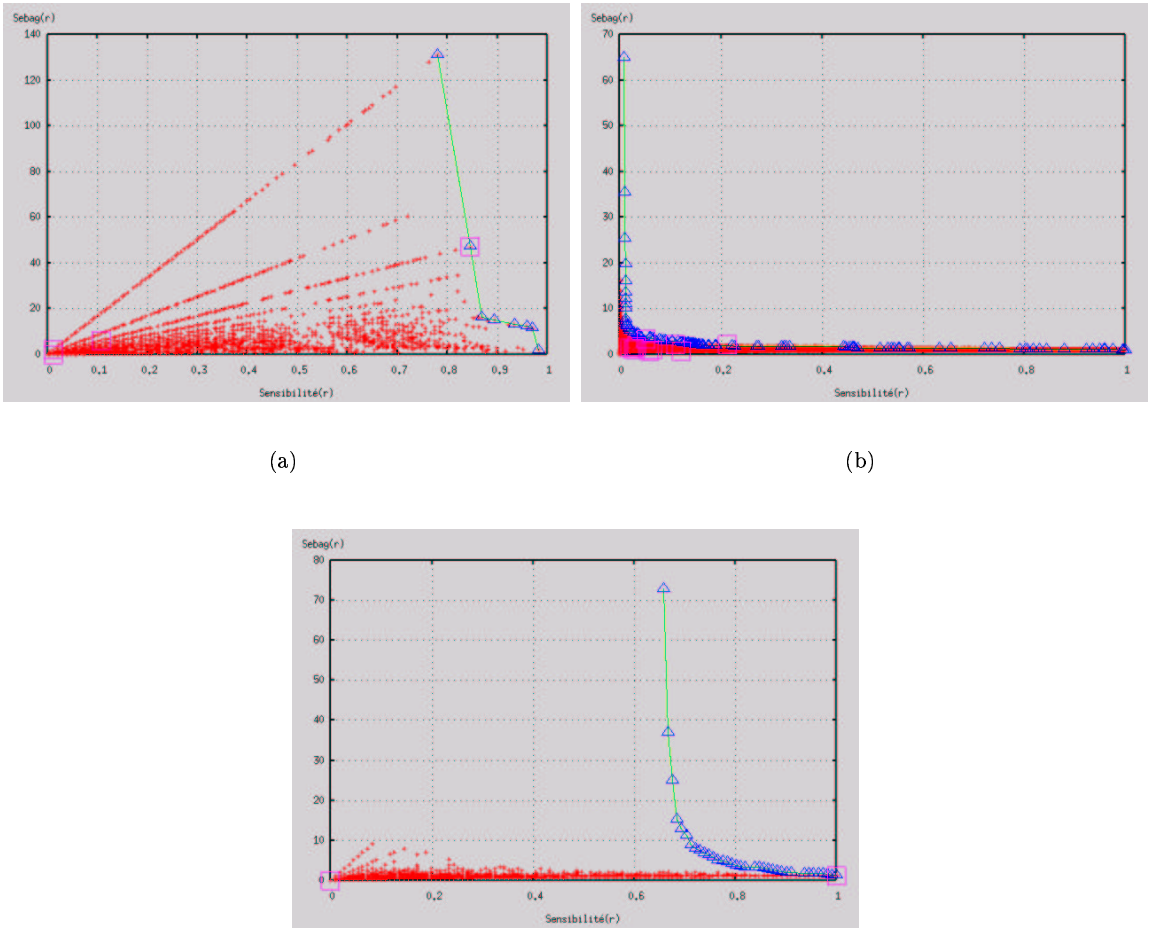
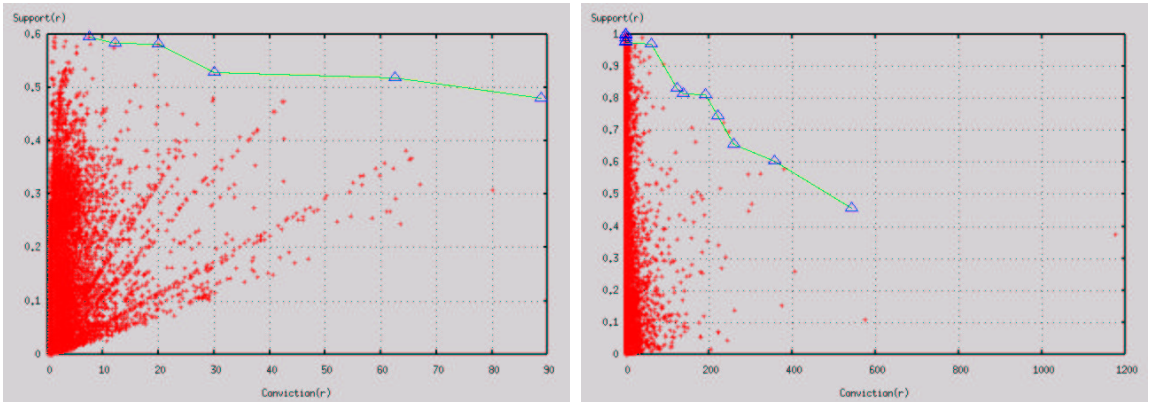


FIG. 5.12 – *Règles simples : Sensibilité versus Sebag sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs. Carrés : algorithme d'induction d'arbres de décision.*

et ce par le biais d'un filtrage.

Concernant le couple $(Sensibilité, Sebag)$ à la figure 5.16 : avec ce couple également, il n'y a pas d'antagonisme lorsque des règles généralisées sont envisagées sur le jeu de données « Abonnés ».



(a) (b)

FIG. 5.13 – *Règles généralisées : Conviction versus Support sur les jeux de données « Vote » (a) et « Abonnés » (b). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs.*

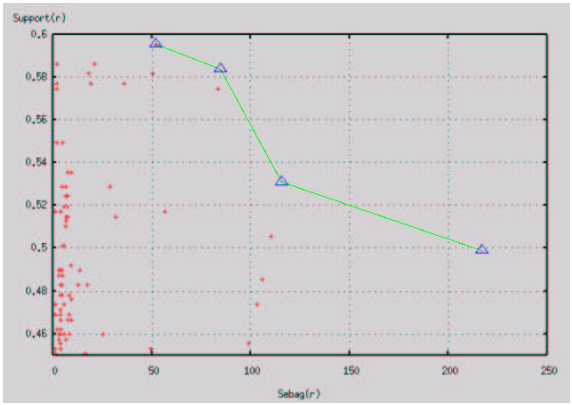


FIG. 5.14 – *Règles généralisées : Sebag versus Support sur le jeu de données « Vote ». Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs.*

5.4.4 Résultats : règles

Cette section présente les résultats que nous avons obtenus lorsque l'on considère une approche selon les règles. Elle se focalise sur deux aspects : d'une part, la mise en évidence de certains attributs au sein des règles trouvées par l'algorithme génétique, attributs qui ne sont pas présents simultanément dans les règles déterminées par les méthodes standards. D'autre part, cette section met en lumière la variation des résultats selon le choix des mesures permettant d'extraire les règles.

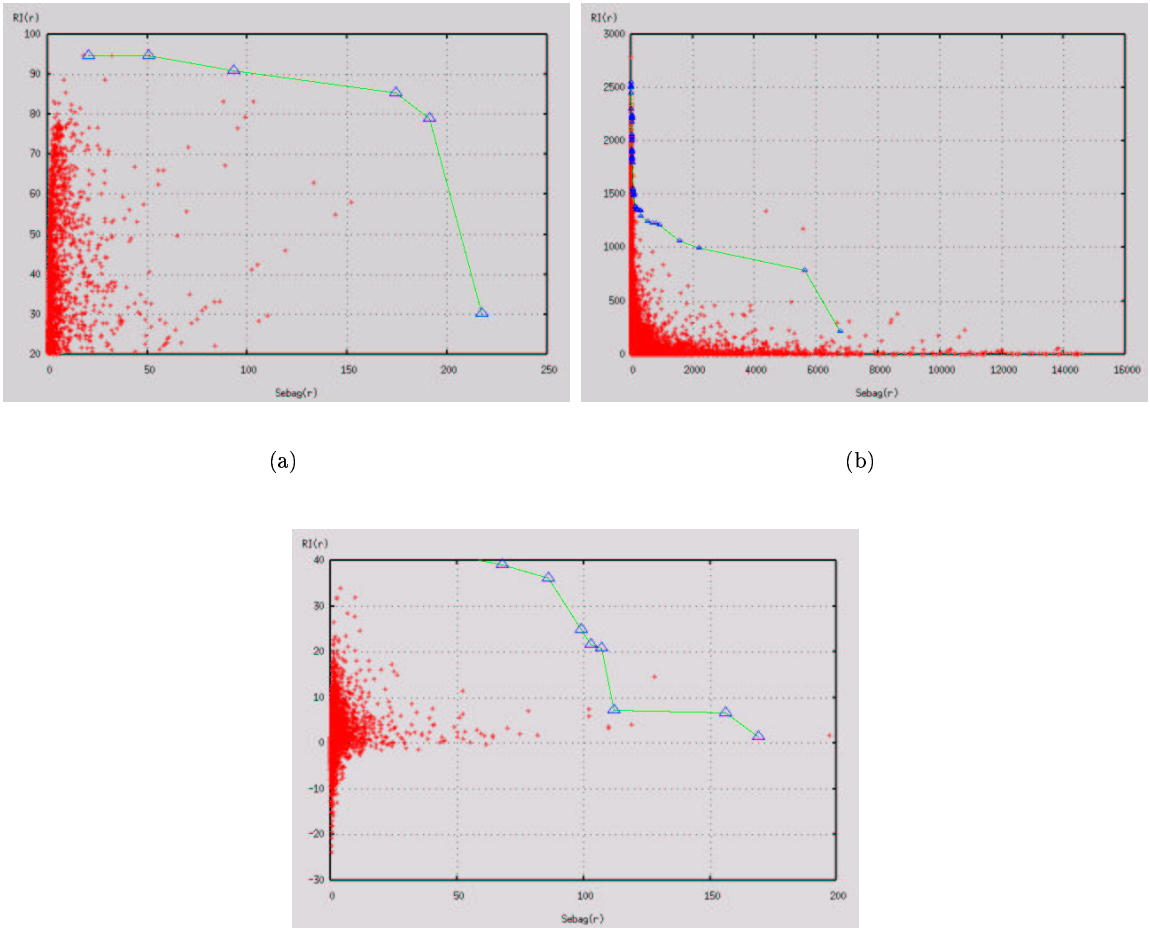


FIG. 5.15 – *Règles généralisées : Sebag versus RI sur les jeux de données « Vote » (a), « Abonnés » (b) et « Sonar » (c). Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs.*

Ensemble de compromis

Comme nous l'avons évoqué précédemment, l'approche présentée se focalise uniquement sur l'optimisation de critères de qualité objectifs basés sur des mesures statistiques. On obtient par cette méthode un ensemble de règles optimisant simultanément, au mieux, deux (ou plus) critères choisis par l'utilisateur. On remarque souvent que les règles ainsi obtenues ne sont pas découvertes par les algorithmes standards ce qui est naturel puisqu'ils utilisent des critères différents pour parcourir l'espace de recherche et pour sélectionner les règles. Un des intérêts de cette méthode basée sur l'algorithme génétique multi-objectifs réside dans la possibilité offerte par la fonction d'évaluation

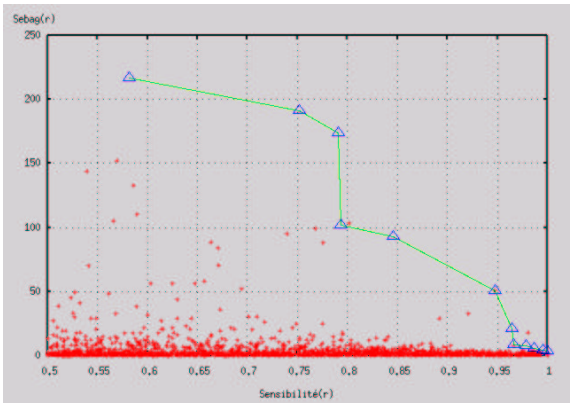


FIG. 5.16 – *Règles généralisées : Sensibilité versus Sebag sur le jeu de données « Vote ». Croix : tirage aléatoire, triangles : algorithme génétique multi-objectifs.*

d'adapter la recherche à plusieurs critères choisis et de les faire varier d'une exécution à l'autre. On opère de cette façon un parcours plus large de l'espace de recherche permettant d'extraire des règles. Les règles ainsi obtenues doivent ensuite être filtrées selon des critères d'intérêt liés au domaine et des points de vue subjectifs.
En effet, ces règles, bien qu'étant des compromis, peuvent représenter des points extrêmes sur la frontière de Pareto auquel cas une ou plusieurs des mesures qui la caractérisent s'en trouvent lésées.

Observons par exemple deux règles obtenues sur le jeu de données « Abonnés » pour le couple $(Sebag, RI)$ (cf. figure 5.10). Ces deux règles ne sont pas trouvées par l'algorithme d'induction d'arbres de décision. Cette première règle est la règle la plus à gauche en haut sur la frontière, donc ayant une valeur de $Sebag$ assez faible, soit environ 1.69428 :

*(I) « **SI** le nombre de semaines restantes est inf. ou égal à 2 **ET***
*l'ancienneté du contrat est inf. ou égal à 27 mois **ET***
*le nombre de jours depuis le dernier changement est inf. ou égal à 697 **ET***
le nombre de correspondants entrants différents par mois est inf. ou égal à 76
***ALORS** l'abonné demande une résiliation de contrat. »*

La règle suivante est une règle prise sur le « milieu » de la frontière de Pareto :

*(II) « **SI** l'âge de l'abonné est inf. ou égal à 67 ans **ET***
*le nombre de semaines restantes est inf. ou égal à 2 **ET***
*l'ancienneté du contrat est inf. ou égal à 27 mois **ET***
*le nombre de jours depuis le dernier changement est inf. ou égal à 697 **ET***
*le % des appels sortants vers un poste fixe est inf. ou égal à 81.3% **ET***
*le nombre moyen d'appels vers service client par mois est sup. ou égal à 0.25 **ET***

le nombre moyen d'appels à info. conso. par mois est sup. ou égal à 1 ***ET***
le nombre de correspondants entrants différents par mois est inf. ou égal à 47
ALORS *l'abonné demande une résiliation de contrat. »*

Sa valeur de RI est 2.51807 et sa valeur de $Sebag$ est 315. On peut observer que l'antécédent de cette règle *(II)* qui est plus longue que la règle *(I)*, reprend en partie l'antécédent de cette dernière. On obtient aussi une valeur supérieure pour $Sebag$. On a donc ici une règle plus locale et de meilleures performances selon RI et $Sebag$. Si on appliquait un critère lié à la présence d'un attribut particulier comme l'âge de l'abonné, la règle *(I)* serait ignorée.

On peut aussi utiliser un ou plusieurs critères syntaxiques comme par exemple la longueur des règles ou encore la présence de certains attributs. Cette première règle plus longue est la plus à gauche en haut sur la frontière de Pareto, donc pour une valeur de *Sensibilité* faible, possède une valeur de *Sensibilité* de 0.45946 et une valeur de *Spécificité* de 1. Son antécédent reprend plusieurs des termes de l'antécédent *(I)* excepté sur l'attribut A_{47}

$$\textit{(I) «}\ \textbf{SI}\ (A_7 \geq 0.01)\ \textbf{ET}\ (A_9 \geq 0.15)\ \textbf{ET}\ (A_{11} \geq 0.15)\ \textbf{ET}\ (A_{12} \geq 0.17)\ \textbf{ET}\ (A_{15} \geq 0.06)\ \textbf{ET}\ (A_{17} \leq 0.7)\ \textbf{ET}\ (A_{18} \leq 0.65)\ \textbf{ET}\ (A_{25} \geq 0.09)\ \textbf{ET}\ (A_{47} \geq 0.06)\ \textbf{ALORS}\ Class = Mine\ \textit{»}$$

et la règle *(II)* suivante est une règle prise sur le « milieu » de la frontière de Pareto :

$$\textit{(II) «}\ \textbf{SI}\ (A_{11} \geq 0.15)\ \textbf{ET}\ (A_{12} \geq 0.17)\ \textbf{ET}\ (A_{15} \geq 0.06)\ \textbf{ET}\ (A_{47} \geq 0.04)\ \textbf{ALORS}\ Class = Mine\ \textit{»}$$

Résultats guidés par les objectifs

Le choix des mesures influe sur les résultats obtenus. Ainsi, suivant l'objectif de l'utilisateur, des mesures différentes ne donneront bien entendu pas les mêmes résultats. Pour illustrer cet aspect, nous avons considéré deux cas. Celui des règles rares et celui des règles générales nécessitant toutes deux la prise en compte de plusieurs mesures de qualité. Concernant les règles rares, si on considère les deux types de règles rares vues au chapitre 3 : une règle $A \rightarrow C$ est rare si $|A|$ est petit et :

- $|C|$ est « petit » : A recouvre alors presque totalement C.
- ou $|C|$ est « grand » : ce cas peut se produire lorsque quelques règles globales recouvrent presque totalement une classe C, et plusieurs règles locales permettent d'affiner la couverture de C.

Pour extraire de telles règles via une approche multi-objectifs, il est nécessaire de minimiser dans les deux cas le nombre d'instances couvertes, c'est-à-dire de minimiser le *Support*.

Pour les règles rares du premier type, il faut également :

- Maximiser la couverture de C par A c'est-à-dire maximiser la $Confiance$.
- Maximiser la couverture de A par C c'est-à-dire maximiser la $Sensibilité$.

Pour les règles rares du second type, il faut également :

- Maximiser la couverture de C par A c'est-à-dire maximiser la $Confiance$.
- Minimiser la couverture de A par C c'est-à-dire minimiser la $Sensibilité$.

Nous avons donc considéré trois mesures dans l'algorithme génétique : le *Support* à minimiser, la $Confiance$ à maximiser et enfin, la $Sensibilité$ à maximiser pour la recherche de règles rares du premier type, et à minimiser pour celles du second type. Ceci a été effectué sur le jeu de données « Sonar » pour la découverte de règles rares appartenant à la première catégorie. Voici un exemple de règle obtenue :

« ***SI*** $(A_{10} \geq 0.5378)$ ***ET*** $(A_{21} \geq 0.9777)$ ***ET*** $(A_{46} \geq 0.4232)$
ALORS $(A_1 \geq 0.115)$ ***ET*** $(A_{14} \leq 0.1926)$ ***ET*** $(A_{19} \leq 0.4474)$ ***ET*** $(A_{54} \geq 0.0022)$ »

Cette règle a un *Support* de 0.00481, une $Confiance$ de 1 et une $Sensibilité$ de 1 également. Elle n'est pas trouvée par l'algorithme APRIORI. Elle peut représenter un intérêt du fait de la longueur de son conséquent.
De même, pour la découverte de règles générales, il est utile de prendre en compte plusieurs mesures. En effet, une règle peut être générale mais hélas, ne présenter que peu d'intérêt pour l'utilisateur car triviale pour ce dernier. Il convient alors d'envisager une ou plusieurs autres mesures permettant de mesurer diverses qualité(s) que la règle doit satisfaire au risque d'être inutile, le cas échéant. Nous présentons ici un exemple de règle obtenue sur le jeu de données « Vote ». L'algorithme génétique a été exécuté pour maximiser le *Support* des règles, ainsi que les mesures *Sebag* et *RI*. Maximiser le *Support* permet de favoriser l'émergence de règles générales, et maximiser les deux autres mesures permet de maximiser d'une part le ratio de vrais positifs par rapport au vrais négatifs (rôle de *Sebag*) et d'autre part de maximiser la connaissance a posteriori sur C par rapport à la connnaissance a priori sur C (rôle de *RI*). Voici cette règle :

« ***SI*** *export-administration-act-south-africa=y* ***ET***
Class=democrat
ALORS *physician-fee-freeze=n.* »

Cette règle a un *Support* de 0.56558, une valeur de *Sebag* de 27.33333 et de valeur de *RI* de 94.75862. Cette règle n'est pas non plus trouvée par l'algorithme APRIORI.
Ainsi les algorithmes génétiques constituent un des outils pour atteindre des objectifs différents dans le sens où il est juste nécessaire de changer l'ensemble de mesures considéré pour aboutir au résultat escompté. La démarche illustrée ici pour trois critères pouvant bien entendu être étendue à un nombre beaucoup plus important de critères.

5.4.5 Sensibilité des paramètres de l'algorithme génétique

Le paramétrage de l'algorithme génétique multi-objectifs est un point crucial et cette section explore cet aspect car on peut s'interroger sur l'influence des paramètres sélectionnés sur les résultats de l'algorithme génétique.
Outre le nombre d'individus dans la population, le taux de croisement et le taux de mutation, la distance d'influence propre à notre algorithme génétique mutli-objectifs et intervenant dans la sélection des individus constitue un paramètre clé. Cette section explore les différents résultats obtenus selon les valeurs de ces paramètres.
Nous présentons ici, l'influence que peuvent induire les variations des paramètres sur l'échantillonnage de la frontière de Pareto, c'est-à-dire sur la densité des points découverts sur cette frontière.
A titre d'exemple, nous avons choisi le couple $(Sensibilité, Spécificité)$ sur le jeu de données « Sonar ». Cependant nous avons réalisé le même type d'expériences sur plusieurs jeux de données et pour les couples de mesures que nous avons présentés dans cette thèse. Ces différents paramétrages sont rassemblés dans les tableaux 5.2 pour les règles simples et 5.3 pour les règles généralisées. Nous présentons dans cette section des valeurs « charnières » autour desquelles les résultats obtenus se détériorent. Nous avons constaté de façon empirique que ces paramétrages sont propres à chaque jeu de données mais ne sont pas sensibles aux mesures considérées. Nous illustrons donc ceci sur le jeu de données « Sonar » et pour chaque paramètre. Le graphe représente le nuage aléatoire obtenu sur le jeu de données correspondant ainsi que plusieurs frontières de Pareto engendrées suite à l'exécution de l'algorithme génétique pour diverses valeurs de ce paramètre.

Variation de la taille de la population. Le graphe 5.17 présente la variation de la forme de la frontière de Pareto relativement à la variation du nombre d'individus dans la population. Nous rappelons que l'algorithme génétique utilisé est basé sur une population de taille fixe. Nous avons effectué quatre exécutions de l'algorithme génétique pour les tailles de 30, 100, 200 et 300 individus. On peut constater sur ce graphe l'effet du nombre d'individus sur la forme de la frontière de Pareto : plus le nombre d'individus dans la population est élevé (et ce jusqu'à une limite approximative de 200 individus), plus la frontière est diversifiée et constituée de points non trouvés par la méthode du tirage aléatoire. Au delà de ce seuil, la frontière est constituée de moins bons compromis comme on peut le remarquer dans le graphe.

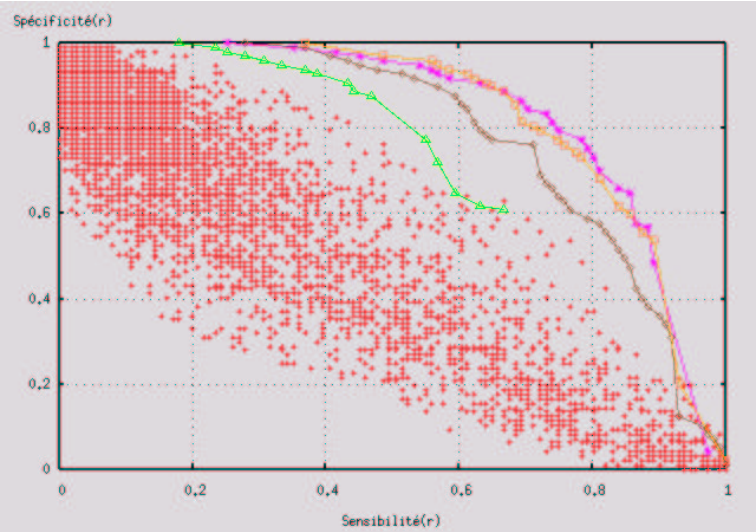


FIG. 5.17 – *Impact de la variation de la taille de la population sur la forme de la frontière de Pareto. Triangles verts : 30 individus. Losanges marrons : 100 individus. Etoiles violettes : 200 individus. Carrés oranges : 300 individus.*

Variation du taux de croisement. Comme illustré dans le graphe de la figure 5.18, la variation du taux de croisement, qui permet rappelons-le, d'effectuer un parcours de l'espace de recherche, modifie la frontière de Pareto de la façon suivante : au delà d'un taux de 80%, la frontière n'est plus améliorée. Les trois frontières de Pareto représentées ont été obtenues respectivement pour des taux de croisement valant 10%, 80% et 100%. Au delà de 80%, les résultats sont peu sensibles. L'algorithme est relativement robuste vis à vis de ce paramètre.

Variation du taux de mutation. Le taux de mutation, comme nous l'avons déjà évoqué, permet d'apporter de la diversité au niveau des solutions trouvées par l'algorithme génétique. La variation de ce taux induit une détérioration de la frontière de Pareto lorsqu'il est augmenté ou diminué par rapport à une valeur « charnière » valant l'inverse du nombre d'attribut N du jeu de données. Ceci est représenté dans le graphe de la figure 5.19 pour trois exécutions différentes de l'algorithme génétique avec le taux de mutation (TM) valant 0.4%, $\frac{1}{N}$ et 40%.

Variation de la distance d'influence. L'augmentation de la distance d'influence a pour impact une déterioration de la forme de la frontière de Pareto comme le présente le graphe de la figure 5.20 au travers de trois frontières de Pareto obtenues pour les distances d'influence suivantes 0.0016, 0.0125 et 0.5. Ici aussi, et pour ce jeu de données, si on s'éloigne de la distance d'influence $\delta = 0.0125$ déterminée empiriquement, on obtient une détérioration de la frontière de Pareto.

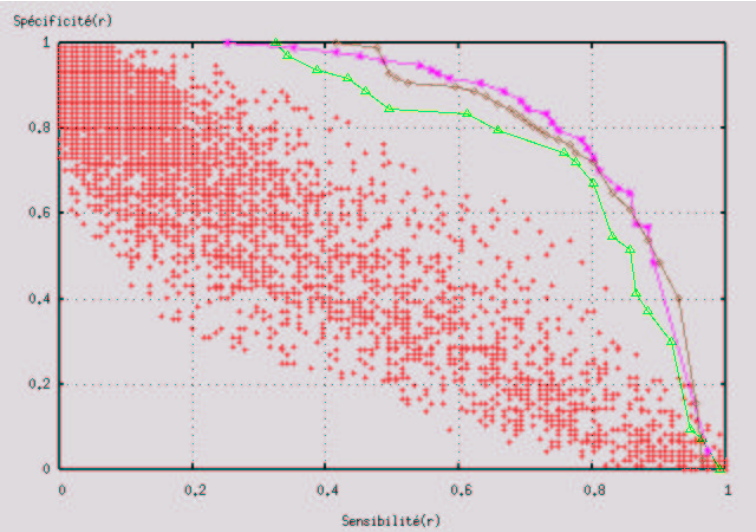


FIG. 5.18 – *Impact de la variation du taux de croisement sur la forme de la frontière de Pareto. Triangles verts : taux de croisement (TC) :* 10%. *Etoiles violettes :* $TC = 80\%$. *Losanges marrons :* $TC = 100\%$.

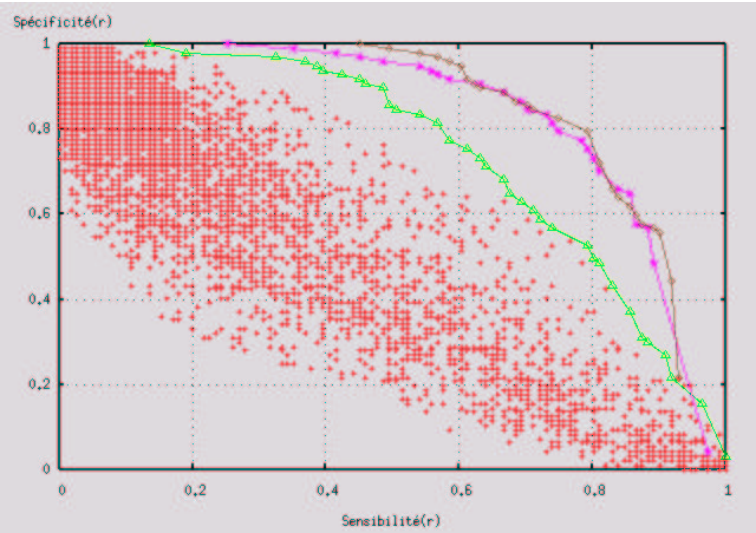


FIG. 5.19 – *Impact de la variation du taux de mutation sur la forme de la frontière de Pareto. Triangles verts : taux de mutation (TM) :* 0.4%. *Etoiles violettes :* $TM = \frac{1}{N}$. *Losanges marrons :* $TM = 40\%$.

Conclusion. Cette section a mis en lumière l'impact de la variation des paramètres sur la qualité de la frontière de Pareto au travers d'une heuristique qui est illustrée ici

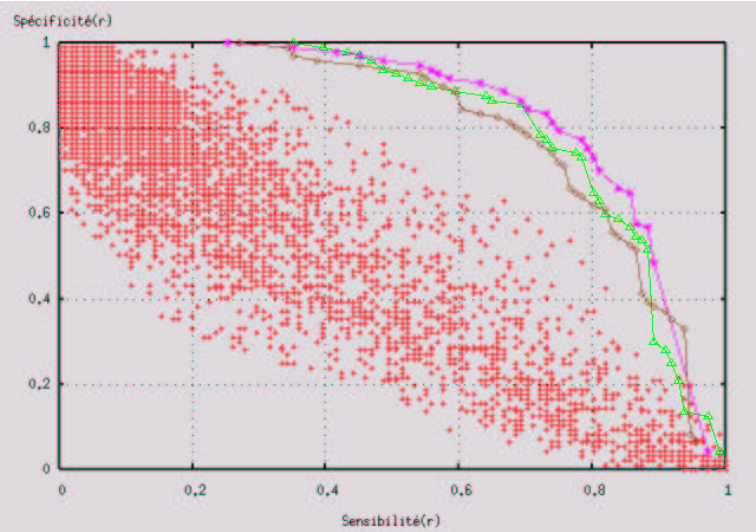


FIG. 5.20 – *Impact de la variation de la distance d'influence sur la forme de la frontière de Pareto. Triangles verts :* $\delta = 0.0016$. *Etoiles violettes :* $\delta = 0.0125$. *Losanges marrons :* $\delta = 0.5$.

sur un jeu de données particulier. Il ressort qu'il est nécessaire de paramétrer l'algorithme génétique relativement à des valeurs charnières déterminées de façon empirique pour la taille de la population, les taux de croisement et de mutation et enfin pour la distance d'influence. Ces seuils sont a priori insensibles aux mesures considérées. Les tests ne sont faits ici que sur un jeu de données. De fait, il faudrait réaliser une méta-optimisation multi-critères. Ces résultats, peut-être sujets à caution, confirment une relative robustesse des algorithmes génétiques vis à vis de leurs paramètres.

5.4.6 Optimisation de l'évaluation des mesures de qualité

La principale faiblesse des algorithmes évolutionnaires en fouille de données est relative à la quantité de ces mêmes données. En effet, plus le jeu de donnée est volumineux, plus l'évaluation d'un individu est coûteuse et l'exécution longue. Notons toutefois, que pour l'utilisateur final, ce type d'algorithme ne doit pas nécessairement être exécuté de nombreuses fois, ce qui rend toute relative la notion de coût liée à l'évaluation des individus.
On cite fréquemment deux principales approches pour accroître la vitesse d'exécution des algorithmes génétiques en fouille de données lorsque la quantité de celles-ci est très grande :

- La première consiste à échantillonner le jeu de données pour évaluer un individu. Ce sous-ensemble peut-être obtenu soit par le biais d'une sélection aléatoire d'un certain nombre d'instances du jeu initial avant l'exécution. Soit par une sélection

adaptative, au cours de l'exécution, d'instances du jeu [Bhat98].

- La seconde consiste à paralléliser l'algorithme [Frei98c].

Afin d'optimiser l'évaluation des individus de la population, nous avons expérimenté une approche qui n'a ni recours à l'échantillonnage ni à la parallélisation. Celle-ci consiste à « garder une trace » en mémoire centrale et pendant une durée variable, des individus générés par l'algorithme au cours de son exécution. Cette structure temporaire permet de stocker au travers d'un arbre n-aire les valeurs des critères des individus créés afin d'éviter le parcours coûteux du jeu de données. La grande quantité de règles engendrées pendant l'exécution de l'algorithme et devant être stockées peut provoquer un dépassement de mémoire qui est évité par la prise en compte de l'âge des règles. Seules les plus récentes sont gardées en mémoire, les plus anciennes étant éliminées (uniquement si le besoin s'en fait sentir, c'est-à-dire si les dimensions de la structure mises à jour de façon dynamique dépassent les limites fixées au préalable). La figure 5.21 présente l'aspect de cette structure de données, sur laquelle cette méthode repose, pour l'exemple suivant : on suppose que le jeu de données est constitué de 11 attributs entiers : A_1, A_2, ..., A_{10} et C. Ce dernier attribut constitue le conséquent des règles car nous ne considérons ici que le cas des règles simples. Toutes les règles ont le même conséquent et sont caractérisées par la donnée de deux mesures calculées sur le jeu de données. La figure 5.4 représente la structure après l'ajout successif des règles du tableau 5.4.

Num.	*Règle*	*Crit. 1*	*Crit. 2*
r_1	***SI*** $A_2 = 1$ ***ALORS*** C	C_1	C'_1
r_2	***SI*** $A_5 = 3$ ***ALORS*** C	C_2	C'_2
r_3	***SI*** $A_5 = 3$ ***ET*** $A_8 \geq 1$ ***ALORS*** C	C_3	C'_3
r_4	***SI*** $A_5 = 3$ ***ET*** $A_8 < 8$ ***ALORS*** C	C_4	C'_4
r_5	***SI*** $A_5 = 3$ ***ET*** $A_8 = 2$ ***ET*** $A_9 = 1$ ***ALORS*** C	C_5	C'_5
r_6	***SI*** $A_5 = 1$ ***ET*** $A_8 = 5$ ***ALORS*** C	C_6	C'_6

TAB. 5.4 – *Exemples de 6 règles insérées dans le cache.*

Les règles ayant des sous-ensembles communs de termes consécutifs « partagent » le même parcours dans la structure arborescente. Chaque nœud est relatif à un attribut. Pour un nœud donné, un sous-arbre de type fils aîné-frère droit contient les différentes règles dont ce nœud représente le dernier terme de l'antécédent.

Lors de la création de la population initiale, cette structure est également initialisée par l'ajout de toutes les règles correspondantes. Lorsqu'un individu est généré par l'algorithme suite à un croisement ou une mutation, la recherche de la règle associée à celui-ci est effectuée dans la structure par le biais de l'algorithme de la figure 5.23. Si

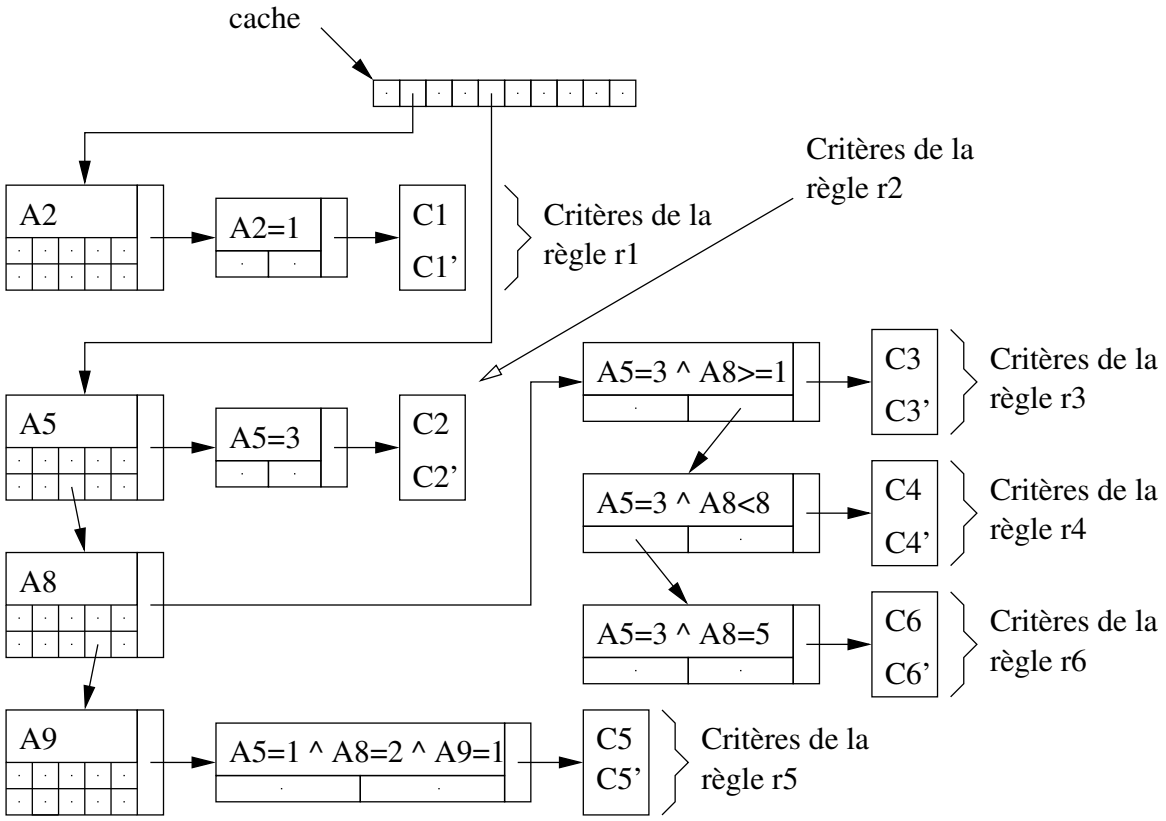


FIG. 5.21 – *Structure du cache pour l'optimisation de l'évaluation des individus de la population de l'algorithme génétique. Ce schéma représente le cache suite à l'insertion des 6 règles du tableau 5.4.*

elle est trouvée, les critères contenus dans l'arbre sont extraits et assignés à l'individu correspondant, sinon l'évaluation a lieu sur le jeu de données.
Le graphe de la figure 5.22 présente le résultat de la mise en œuvre de cette méthode sur le jeu de données « Vote ». Ce graphe met en évidence l'apport de la mise en œuvre d'un cache permettant d'éviter le parcours coûteux du jeu de données lorsqu'un individu de la population de l'algorithme génétique a subi une modification suite à une mutation ou bien à un croisement.

5.5 Conclusion

Dans ce chapitre nous avons présenté le concept d'optimisation multi-objectifs ainsi que les nombreuses méthodes qui lui sont associées. Nous avons abordé les travaux qui nous semblent les plus significatifs en fouille de données faisant intervenir un algorithme évolutionnaire multi-objectifs. Nous avons ainsi mis en évidence que ces travaux sont généralement dédiés à un domaine particulier.

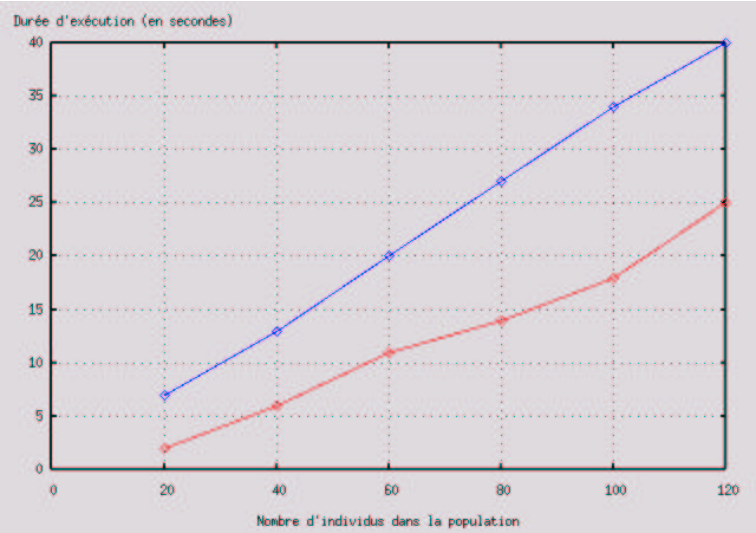


FIG. 5.22 – *Comparatif entre les durées d'exécution de l'algorithme sur le jeu de données « Vote » lorsque la structure de cache est mise en œuvre (courbe rouge en bas) et lorsque l'algorithme génétique doit effectuer un parcours du jeu de données pour chaque individu ayant subi une modification (courbe bleue en haut).*

1) **Algorithme** Recherche d'une règle à partir du cache
2) **Données :**
3) r : la règle recherchée
4) S : le cache
5) C_i : le $i^{ème}$ critère de la règle r recherchée
6) **Début**
7) **Si** la règle r est présente dans le cache S **alors**
8) **Pour** i **de** 1 **à** *nombre de critères* **faire**
9) C_i = extraire le $i^{ème}$ critère de r
10) **Fin pour**
11) Mise à jour de S suivant l'âge des règles
12) **Sinon**
13) Ajouter la règle r dans S
14) **Fin si**
15) **Fin**

FIG. 5.23 – *L'algorithme d'évaluation de règles dans la structure de cache.*

Ce chapitre s'est focalisé sur l'utilisation d'un algorithme génétique multi-objectifs

pour l'extraction de règles de dépendance par le biais de mesures présentant des antagonismes. Nous avons mis en évidence l'apport de cette approche. Notre objectif étant de fournir à une méthode annexe de filtrage, un ensemble de règles qui ne constitue pas le « meilleur modèle » mais qui représente l'ensemble des meilleurs compromis. Notre approche s'est notamment focalisée sur l'intérêt et la qualité des règles de dépendance considérées d'un point de vue général sans critères subjectifs liés au domaine d'application.
Nous avons également abordé deux problèmes majeurs qui sont la sensibilité des paramètres de l'algorithme génétique ainsi que l'optimisation de l'évaluation des mesures de qualité.
Concernant le premier problème, il ressort que le paramétrage de l'algorithme influence sensiblement la nature des résultats obtenus, comme c'est généralement le cas lorsqu'un algorithme évolutionnaire est mis en œuvre. De plus, ce paramétrage est dépendant du jeu de données en présence.
Concernant l'optimisation de l'évaluation des mesures de qualité, nous avons mis en œuvre une méthode basée sur une structure de cache permettant de diminuer le coût lié au parcours du jeu de données lors de l'évaluation d'un individu.

Chapitre 6

Bilan

Sommaire

6.1 Conclusion

Nous avons dans cette thèse abordé la fouille de données sous l'angle de l'optimisation en nous focalisant sur l'approche évolutionnaire pour la découverte de règles de dépendance. Nous avons mis en évidence l'intérêt de cette approche par rapport aux algorithmes standards d'extraction de règles utilisés dans ce domaine. Ces derniers de par leur architecture sont dans l'impossibilité de faire face à certains problèmes tels que la prise en compte de plusieurs critères de sélection simultanément.

Nous avons présenté les résultats de notre approche sur divers jeux de données pour lesquels certaines mesures de qualité sont liées par des antagonismes. Nous avons considéré dans ce mémoire deux types de règles : d'une part les règles *simples* qui sont constituées d'une seule condition dans leur conséquent et d'autre part, les règles *généralisées* qui ont plusieurs conditions dans leur conséquent. Notre objectif s'est focalisé sur l'étude des espaces de recherche des mesures associés à ces deux types de règles sur divers jeux de données. Nous avons donc présenté un certain nombre de mesures ainsi que plusieurs travaux relatifs à celles-ci. Nous avons pu voir que ces travaux tendent généralement à formaliser des principes caractérisant les mesures d'intérêt. Diverses voies ont été explorées dans la littérature comme celles consistant à faire varier les critères de choix, ce qui n'apporte pas de solutions concernant leur application. D'autres tentent de satisfaire par une règle, plusieurs critères au travers de l'optimisation d'une unique mesure. Ces approches ont motivé nos choix notamment celui de ne pas aborder

le problème sous un angle analytique. En effet, le comportement des mesures est très nettement influencé par les jeux de données sur lesquels elles sont appliquées, rendant par conséquent une telle approche inadaptée.

Nous avons étendu ces travaux selon divers points de vue notamment en ce qui concerne les espaces de recherche. En effet, nous envisageons la fouille de données d'un point de vue optimisation ce qui nécessite l'étude des espaces de recherche des mesures à optimiser. Nous avons présenté diverses approches permettant de mettre en lumière des particularités liées aux mesures sur certains jeux de données, et qui justifient l'approche expérimentale que nous avons choisie. Ces particularités sont relatives à l'existence d'un nombre restreint de règles ayant une valeur élevée pour une mesure donnée, de même qu'il existe des règles assez longues et « bonnes » relativement à une mesure. Nous avons également mis en évidence par la variation de la syntaxe des règles et de la distance de Hamming (par rapport à une règle de l'algorithme d'induction d'arbres de décision) l'existence de maxima locaux. Enfin, nous avons montré de façon expérimentale l'existence d'antagonismes entre certaines mesures.

L'ensemble de ce travail constitue une base pour la mise en œuvre de techniques évolutionnaires selon deux axes : celui de la prise en compte d'une seule mesure, et celui de la prise en compte d'un ensemble de mesures. Nous avons présenté la métaheuristique des algorithmes évolutionnaires d'une façon générale ainsi que ses adaptations dans le contexte de la fouille de données. Nous avons ensuite présenté un état de l'art des travaux existants mettant en œuvre ces algorithmes en fouille de données. Nous nous démarquons des principaux travaux actuels par l'usage d'un algorithme génétique basé sur une mesure de qualité pour la découverte de règles de dépendance lorsque les espaces de recherche présentent des optima locaux que les méthodes standards ne sont pas à même d'appréhender.
Les résultats que nous avons obtenus sur différents jeux de données, nous ont permis de conclure que dans certains cas, les algorithmes génétiques permettent d'identifier des solutions ignorées par des méthodes plus traditionnelles. Notre approche expérimentale permet notamment d'apporter des solutions lorsque qu'il existe de « bonnes » règles de dépendance selon une mesure mais qui sont ignorées par les techniques standards car trop longues. D'autre part, certaines règles non envisagées par ces mêmes techniques peuvent également être découvertes. Enfin, nous avons abordé le cas de mesures antagonistes qui nécessitent une approche multi-objectifs pour obtenir des modèles intéressants.

La première partie de notre approche, a donc été relative à la prise en compte d'une seule mesure pour la recherche de règles de dépendance par le biais d'un algorithme génétique. La seconde partie a consisté à utiliser cette même technique dans le cas de plusieurs mesures de qualité (et en particulier de deux). Nous avons mis en lumière l'aptitude des algorithmes évolutionnaires à déterminer un ensemble de bons

compromis, c'est-à-dire de « bonnes » règles et ce relativement aux mesures considérées lorsque celles-ci sont liées par un antagonisme. Nous avons abordé les travaux qui nous semblent les plus significatifs en fouille de données basés sur un algorithme évolutionnaire multi-objectifs. Nous avons ainsi mis en évidence que ces travaux sont généralement dédiés à un domaine particulier.
Nous nous sommes focalisés sur l'utilisation d'un algorithme génétique multi-objectifs pour l'extraction de règles de dépendance par le biais de mesures présentant des antagonismes. Nous avons mis en évidence l'apport de cette approche. Notre objectif étant de fournir à une méthode annexe de filtrage, un ensemble de règles qui ne constitue pas le « meilleur modèle » mais qui représente l'ensemble des meilleurs compromis. Notre approche s'est notamment focalisée sur l'intérêt et la qualité des règles de dépendance considérées d'un point de vue général sans critères subjectifs liés au domaine d'application. Nous avons également abordé deux problèmes qui sont la sensibilité des paramètres de l'algorithme génétique ainsi que l'optimisation de l'évaluation des mesures de qualité.

6.2 Perspectives

A la lumière de ce qui a été abordé dans ce mémoire, cette section présente plusieurs voies à explorer dans la continuation de nos travaux.

Mesures. Nous nous sommes focalisées dans le chapitre 3 sur la considération et l'étude de mesures objectives principalement. Il semble intéressant d'envisager le cas de mesures subjectives, notamment au travers d'un algorithme évolutionnaire interactif. Ceci implique de connaître des « avis » d'experts des domaines considérés. Concernant les mesures objectives, malgré le fait que le sujet ait été abordé assez souvent, une étude systématique et exhaustive des mesures existantes s'impose afin de permettre le choix d'une ou d'un ensemble de celles-ci en vue du processus de fouille de données. Choisir une mesure parmi les nombreuses mesures existantes n'est pas toujours aisé, c'est pourquoi il serait intéressant d'intégrer une solution multi-critères du type de celle développée par [Lenc02] pour le choix d'un ensemble de mesures.

Elaboration de filtre pour le post-traitement. Comme nous l'avons vu à plusieurs reprises dans l'analyse des règles, notamment dans le cas des règles généralisées qui sont associées à des espaces de recherche beaucoup plus grands que ceux des règles simples, le nombre de règles fourni par l'algorithme génétique est souvent élevé ce qui pose un problème quant à son exploitation par l'utilisateur final. Des méthodes de filtrage des résultats sont nécessaires et nous avons implémenté et mis en œuvre des filtres simples dans le cadre de l'application d'algorithme génétique mono-objectif. Cependant cet aspect constitue une voie d'investigation qui dépasse le cadre de nos travaux puisque, à l'évidence, nous nous sommes concentrés sur la recherche de règles

devant satisfaire des mesures de qualité et non sur la recherche d'un modèle compact, précis et intéressant.

Optimisation de l'évaluation de la fonction d'adaptation. La méthode présentée dans la section 5.4.6 nécessiterait une extension permettant de prendre en compte « l'âge » des règles qui sont stockées dans la structure de cache. En effet, lors de l'initialisation de la population initiale, de nombreuses règles y sont insérées, cependant, celles-ci sont pour la plupart de mauvaise qualité et, de fait, ne devraient pas rester dans le cache car l'algorithme génétique faisant converger la population vers un ensemble de bons individus, il est peu problable que l'algorithme ait recours à la recherche de ces règles ultérieurement dans le cache. Ceci permettrait notamment d'éviter que la mémoire centrale ne soit saturée par les ajouts successifs de règles durant l'exécution de l'algorithme.

Table des figures

Liste des tableaux

Bibliographie

[Agra93a] R. Agrawal, T. Imielinski, and A. Swami. Database mining : a performance perspective. *IEEE Transaction on Knowledge and Data Engineering : Special issue on Learning and Discovery in Knowledge-Based Databases*, 5(6) :914–925, december 1993.

[Agra93b] R. Agrawal, T. Imielinski, and A. Swani. Mining association rules between sets of items in large databases. In ACM Press, editor, *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD'93)*, pages 207–216. Washington DC, USA, may 1993.

[Agra94] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94)*, pages 428–499. Morgan Kaufmann, september 1994.

[Agra95] R. Agrawal and R. Srikant. Mining sequential patterns. In IEEE Computer Society Press, editor, *Proceedings of the 11th International Conference on Data Engineering (ICDE'95)*, pages 3–14. March 1995.

[Agra98] R. Agrawal, J. Gehrke, D. Gunopulos, and O. Raghawan. Automatic subspace clustering of high dimensional data for data mining applications. In ACM Press, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'98)*, pages 94–105. June 1998.

[Alba02] E. Alba and M. Tomassini. Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6 :443–462, 2002.

[Ali92] K. M. Ali and M. J. Pazzani. *Reducing the small disjuncts problem by learning probalistic concept descriptions*, volume 3. 1992.

[Angl97] C. Anglano, A. Giordano, G. Lo Bello, and L. Saitta. A network genetic algorithm for concept learning. In *Proceedings of the 7th International Conference on Genetic Algorithms (ICGA'97)*, pages 434–441. 1997.

[Arau99] D.L.A. Araujo, H.S. Lopes, and A. A. Freitas. A parallel genetic algorithm for rule discovery in large databases. In *Proceedings 1999 IEEE Systems, Man and Cybernetics Conference, v. III*, pages 940–945. Tokyo, Japan, october 1999.

[Arau00] D. L. A. Araujo, H. S. Lopes, and A. A. Freitas. Rule discovery with a parallel genetic algorithm. In *Proceedings of International Conference on*

Genetic and Evolutionary Computation (GECCO'00) - Workshop program, pages 89–92. Las Vegas, Nevada, USA, july 2000.

[Augi95] S. Augier, G. Venturini, and Y. Kodratoff. Learning first order logic rules with a genetic algorithm. In *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining (KDD'95)*. Montréal, Canada, 1995.

[Azar96] S. Azarm. Multiobjective optimum design. 1996.

[Banz00] W. Banzhaf. Interactive evolution. *Evolutionary Computation*, 1 :228–236, 2000.

[Baya98] R. J. Bayardo. Efficiently mining long patterns from databases. In ACM Press, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'98)*, pages 85–93. 1998.

[Bhat98] S. Bhattacharrya. Direct marketing response models using genetic algorithms. In AAAI Press, editor, *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 144–148. 1998.

[Bhat99] S. Bhattacharyya. Direct marketing performance modeling using genetic algorithms. *INFORMS Journal of Computing*, 11(3), summer 1999.

[Bhat00] S. Bhattacharyya. Evolutionary algorithms in data mining : multi-objective performance modeling for direct marketing. In AAAI Press, editor, *Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining (KDD'00)*. Boston, USA, august 2000.

[Boja99] C. E. Bojarczuk, H. S. Lopes, and A. A. Freitas. Discovering comprehensible classification rules using genetic programming : a case study in a medical domain. In *Proceedings of the International Conference on Genetic and Evolutionary Computation (GECCO'99)*, pages 953–958. Orlando, Floride, USA, july 1999.

[Boja00] C. E. Bojarczuk, H. S. Lopes, and A. A. Freitas. Genetic programming for knowledge discovery in chest pain diagnosis. *IEEE Engineering in Medecine and Biology magazine - Special issue on data mining and knowledge discovery*, 19(4) :38–44, july/august 2000.

[Bono88] E. Bonomi and J.-L. Lutton. Le recuit simulé. *Pour la Science*, (129) :68–77, july 1988.

[Bran86] J. P. Brans, Ph. Vincke, and B. Mareschal. How to select and how to rank projects : the promethee method. *European journal of Op. Research*, 24 :228–238, 1986.

[Brei84] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and regression trees*. Monterey, California, USA : Wadsorth International Group, 1984.

[Brin97] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In ACM Press, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data (ACM SIGMOD'97)*, pages 255–264. Tucson, Arizona, USA, 13-15 may 1997.

[Carp93] C. Carpineto and G. Romano. Galois : an order-theoretic approach to conceptual clustering. In *Proceedings of the 10th International Conference on Machine Learning (ICML'90)*, pages 33–40. July 1993.

[Carv99] D. R. Carvalho, B. C. Avila, and A. A. Freitas. A hybrid genetic algorithm/decision tree approach for coping with unbalanced classes. In *Proceedings of the 3rd International Conference on the Practical Applications of Knowledge Discovery and Data Mining (PADD'99)*, pages 61–70. London, UK, april 1999.

[Carv00a] D. R. Carvalho and A. A. Freitas. A genetic algorithm-based solution for the problem of small disjuncts. In Springer-Verlag, editor, *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'00) - Lecture notes in Artificial Intelligence 1910*, pages 345–352. Lyon, France, october 2000.

[Carv00b] D. R. Carvalho and A. A. Freitas. A hybrid decision tree/genetic algorithm for coping with the problem of small disjuncts in data mining. In *Proceedings of International Conference on Genetic and Evolutionary Computation (GECCO'00)*, pages 1061–1068. Morgan Kaufmann, Las Vegas, NV, USA, july 2000.

[Carv01] D. R. Carvalho and A. A. Freitas. An immunological algorithm for discovering small-disjuncts rules in data mining. In *Proceedings of the Graduate Student Workshop held at the Genetic and Evolutionary Computation Conference (GECCO'01)*, pages 401–404. San Francisco, California, USA, 2001.

[Carv02a] D. R. Carvalho and A. A. Freitas. A genetic algorithm for discovering small-disjuncts rules in data mining. *To appear in Applied Soft Computing Journal*, 2002.

[Carv02b] D. R. Carvalho and A. A. Freitas. A genetic algorithm with sequential niching for discovering small-dijunct rules. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'02)*. New York, USA, 2002.

[Catt00] R. Cattral, F. Oppacher, and D. Deugo. Using genetic algorithms to evolve a rule hierarchy, 2000.

[Chee96] P. Cheeseman and J. Stutz. Bayesian classification (autoclass) : theory and results. In AAAI Press, editor, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. U. M. Fayyad and G. Piatetsky-Shapiro and P. Smyth and R. Uthutusamy, 1996.

[Coel98] C. A. Coello. An updated survey of genetic algorithms based multiobjective optimization techniques. Technical report Lania-RD-98-08, Xalapa, Veracruz, Mexico, december 1998.

[Coll01] M. Collard and D. Francisci. Evolutionary data mining : an overview of genetic-based algorithms. In *Proceedings of the 8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'01)*, pages 4–10. Antibes, France, 2001.

[Coll02] Y. Collette and P. Siarry. *Optimisation multiobjectif*. 2002.

[Cong00] C. B. Congdon and E. F. Greenfest. Gaphyl : a genetic algorithm approach to cladistics. In A. Wu, editor, *Proceedings of International Conference on Genetic and Evolutionary Computation (GECCO'00) - Workshop*. 2000.

[Crav94] M. Craven and J. W. Shavlik. Using sampling and queries to extract rules from trained neural networks. In *Proceedings of the 11th International Conference on Machine Learning*, pages 37–45. Morgan Kaufmann, New Brunswick, NJ, 1994.

[Dany93] A. P. Danyluk and F. J. Provost. Small disjuncts in action : learning to diagnose errors in the local loop of the telephone network. In *Proceedings of the 10th International Conference on Machine Learning (ICML'93)*, pages 81–88. 1993.

[Deb99] K. Deb. An overview of multi-objective evolutionary algorithms, may 1999.

[Deb01] K. Deb. *Multi-objective optimization using evolutionary algorithms*. 2001.

[Dhar00] V. Dhar, D. Chou, and F. Provost. Discovering interesting patterns for investment decision making with glower - a genetic learner overlaid with entropy reduction. *Data Mining and Knowledge Discovery Journal*, 4(4) :251–280, 2000.

[Emma01] C. Emmanouilidis. Evolutionary multi-objective feature selection and roc analysis with application to industrial machinery fault diagnosis. In *Eurogen 2001*. 2001.

[eo] Eo. Home page : http ://www.cs.waikato.ac.nz/ml/weka.

[Esch] H. Eschenauer, J. Koski, and A. Osyczka. *Multicriteria design optimization : procedures and applications*. 1990.

[Fayy96a] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge discovery and data mining : towards a unifying framework. In AAAI Press, editor, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pages 82–88. August 1996.

[Fayy96b] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in knowledge discovery and data mining*. 1996.

[Fide00] M. V. Fidelis, H. S. Lopes, and A. A. Freitas. Discovering comprehensible classification rules with a genetic algorithm. In *Proceedings of Congress on*

Evolutionary Computation (CEC'00), pages 805–810. La Jolla, California, USA, july 2000.

[Floc95] I. W. Flockart and N. J. Radcliffe. Ga-miner : parallel data mining with hierarchical genetic algorithms - final report. In *EPCC AIKMS GA-Miner - Report 1.0*. Edimbourg University, Canada, november 1995.

[Fons93] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization : formulation, discussion and generalization. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 416–423. San Mateo, California, USA, 1993.

[Fons00] C. M. Fonseca and P. J. Fleming. Multiobjective optimization. *Evolutionary Computation*, 2 :25–37, 2000.

[Fran02] D. Francisci. Analyse de critères et de fonctions de fitness - mise en évidence de surfaces de pareto. Research report ISRN I3S/RR-2002-27-FR, MECOSI Project, Laboratoire I3S, Sophia Antipolis, France, july 2002.

[Fran03a] D. Francisci. Seconde analyse de critères et de fonctions de fitness sur des bases corrélées et non corrélées - mise en évidence de surfaces de pareto. Research report ISRN I3S/RR-2003-04-FR, MECOSI Project, Laboratoire I3S, Sophia Antipolis, France, february 2003.

[Fran03b] D. Francisci and M. Collard. Multi-criteria evaluation of interesting dependencies according to a data mining approach. In *Proceedings of Congress on Evolutionary Computation (CEC'03)*. Canberra, Australia, 8-12 december 2003.

[Frei98a] A. A. Freitas. A multi-criteria approach for the evaluation of rule interestingness. In WIT Press, editor, *Proceedings International Conference*, pages 7–20. Rio de Janeiro, Brazil, september 1998.

[Frei98b] A. A. Freitas. On objective measures of rule surprisingness. In Springer-Verlag, editor, *Proceedings of the 2nd European Symphosium on Principles of Data Mining and Knowledge Discovery (PKDD'98) - Lecture notes in Artificial Intelligence*, page 1510. Nantes, France, september 1998.

[Frei98c] A. A. Freitas and S. H. Lavington. *Mining very large databases with parallel processing*. 1998.

[Frei99a] A. A. Freitas. A genetic algorithm for generalized rule induction. In Springer-Verlag, editor, *Proceedings of the 3rd On-Line International Conference on Soft Computing (WSC3) - R. Roy and al. Advances in Soft Computing - Engineering Design and Manufacturing*, pages 340–353. July 1999.

[Frei99b] A. A. Freitas. On rule interestingness measures. *Knowledge-Based Systems journal*, 12(5-6) :309–315, october 1999.

[Frei01] A. A. Freitas. Understanding the crucial role of attribute interaction in data mining. *Artificial Intelligence Review*, 16(3) :177–199, november 2001.

[Frei02] A. A. Freitas. *Data mining and knowledge dicovery with evolutionary algorithms - Natural computing series*. 2002.

[Gelf91] S. B. Gelfand, C. S. Ravishankar, and E. J. Delp. An iterative growing and pruning algorithm for classification tree design. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13 :163–174, 1991.

[Genn90] J. H. Gennari, P. Langley, and D. Fisher. Models of incremental concept formation. *Artificial Intelligence*, 40 :11–61, 1990.

[Gies78] D. P. Giesy. Calculation of pareto optimal solutions to multiple objective problems using threshold of acceptability constraints. *IEEE Trans. On Autom. Contr.*, AC-23(6), december 1978.

[Gior94] A. Giordana, L. Saitta, and F. Zini. Learning disjunctive concepts by means of genetic algorithms. In *Proceedings of the 10th International Conference on Machine Learning (ICML'94)*, pages 96–104. Morgan Kaufmann, 1994.

[Glov86] F. Glover. Future paths for integer programming and links to artificial intelligence. *Comput. And Ops. Res.*, 13(5) :533–549, 1986.

[Glov90] F. Glover. Artificial intelligence, heuristic frameworks and tabu search. *Managerial and decision economics*, 11(5) :365–375, 1990.

[Glov97] F. Glover and M. Laguna. *Tabu search*. Kluwer Academic Publishers, 1997.

[Gold89] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. 1989.

[Good91] R. M. Goodman and P. Smyth. Rule induction using information theory. In MIT Press, editor, *Proceedings of International Conference on Knowledge Discovery in Databases (KDD'91)*. G. Piatetsky-Shapiro and W. J. Frawley, 1991.

[Guil] S. Guillaume, F. Guillet, and J. Philippé. Contribution of the integration of intensity of implication into algorithm proposed by agrawal.

[Guil00] S. Guillaume. *Traitement des données volumineuses - Mesures et algorithmes d'extraction de règles d'association et règles ordinales*. Ph.D. thesis, Université de Nantes, Institut de Recherche en Informatique de Nantes, France, december 2000.

[Gupt95] A. Gupta, V. Harinarayan, and D. Quass. Aggregate-query processing in data warehousing environments. In Morgan Kaufmann, editor, *Proceedings of the 21th International Conference on Very Large Databases (VLDB'95)*, pages 358–369. September 1995.

[Haim75] Y. Y. Haimes, W. A. Hall, and H. T. Freedman. *Multiobjective optimisation in water resources systems*. 1975.

[Han93] J. Han, Y. Cai, and N. Cercone. Data-driven discovery of quantitative rules in relational databases. *IEEE Transaction on Knowledge and Data Engineering*, 5(1) :29–40, february 1993.

[Han96] J. Han and Y. Fu. Exploration of the power of attribute oriented induction in data mining. In AAAI Press, editor, *Advances in Knowledge Discovery and Data Mining*, pages 399–421. U. M. Fayyad and G. Piatetsky-Shapiro and P. Smyth and R. Uthurusamy, 1996.

[Hand01] D. J. Hand, H. Mannila, and P. Smyth. *Principles of data mining*. Adaptive Computation and Machine Learning. J. Wolfskill, 5 Cambridge Center, 4th floor, Cambridge, MA 02142, mit press edition, 2001.

[Hari96] V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. In ACM Press, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'96)*, pages 205–216. June 1996.

[Hart75] A. Hartigan. *Clustering algorithms*. John Wiley and Sons, New York, USA, 1975.

[Hild01] R. J. Hilderman and H. J. Hamilton. Evaluation of interestingness measures for ranking discovered knowledge. In Springer-Verlag, editor, *Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'01) - Lecture notes in Computer Science*, pages 247–259. D. Cheung, G. J. Williams and Q. Li, Hong Kong, China, april 2001.

[Holl75] J. H. Holland. *Adaptation in natural and artificial systems*. 1975.

[Holt89] R. C. Holte, L. E. Acker, and B. W. Porter. Concept learning and the problem of small disjuncts. In *Proccedings of the 17th International Joint Conference on Artificial Intelligence*, pages 813–818. 1989.

[Horn93] J. Horn, N. Nafpliotis, and D. E. Goldberg. Multiobjective optimization using niched pareto genetic algorithm. Technical report IlliGAl Report 93005, Illinois University, Urbana, Illinois, 1993.

[IBM96] International Business Machines IBM. *IBM Intelligent Miner - User's guide*. IBM, 1996.

[Jain99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering : a survey. *ACM Comput. Surv.*, 31 :264–323, 1999.

[Jani93] C. Z. Janikow. A knowledge-intensive genetic algorithm for supervised learning. *Machine Learning*, 13 :189–228, 1993.

[Kary99] G. Karypis, E. H. Han, and V. Kumar. Chameleon : a hierarchical clustering algorithm using dynamic modeling. *Computer*, 32 :68–75, 1999.

[Keen93] R. L Keeny and H. Raiffa. *Decisions with multiple objectives : preferences and value tradeoff*. 1993.

[Lall02] S. Lallich and O. Teytaud. Evaluation et validation de l'intérêt des règles d'association. Research report, Groupe de travail GafoQualitée de l'action spécifique STIC fouille de bases de données, ERIC, Université Lyon 2, France, 2002.

[Lenc02] P. Lenca, P. Meyer, B. Vaillant, and P. Picouet. Aide multicritère à la décision pour évaluer les indices de qualité des connaissances - modélisation des préférences de l'utilisateur. Research report, Groupe de travail Gafo-Qualitée de l'action spécifique STIC fouille de données, département IASC, ENST Bretagne, France, 2002.

[Lin76] J. G. Lin. Multiple-objective problems : Pareto-optimal solutions by proper equality constraints. *IEEE Trasnsactions on Automatic Control*, 5(AC21) :641–650, october 1976.

[Lin97] J. G. Lin. Proper inequality constraints and maximization of index vectors. *Journal of optimisation theory and applications*, 21(4) :505–521, april 1997.

[Liu98] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98) - Plenary Presentation*. New York, USA, 1998.

[Liu99] B. Liu, Y. Ma W. Hsu, and S. Chen. Discovering interesting knowledge using dm-ii. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99) - Industrial Track*. San Diego, California, USA, 15-18 august 1999.

[Liu01] B. Liu, Y. Ma, and C.-K. Wong. Classification using association rules : weaknesses and enhancements. In V. Kumar et al., editor, *Data mining for scientific applications*. 2001.

[Loev47] J. Loevinger. A systematic approach to the construction and evaluation of tests of ability. *Psychological monographs*, 61(4), 1947.

[Lu95] H. Lu, R. Setinio, and H. Liu. Neurorule : a connectionist approach to data mining. In *Proceedings of the 21th International Conference on Very Large Data Bases (VLDB'95)*, pages 478–489. Morgan Kaufmann, september 1995.

[Mann96] H. Mannila and H. Toivonen. Multiple uses of frequent sets and condensed representations. In AAAI Press, editor, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pages 189–194. August 1996.

[Mays94] L.-Y. Maystre, J. Pictet, and J. Simos. *Méthodes multicritères ELECTRE*. 1994.

[Metr53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Introduction of the metropolis algorithm for molecular-dynamics simulation. *Journal of Chemical Physics*, 21 :1087–1092, 1953.

[Mich83] R. S. Michalski. A theory and methodology of inductive learning. In Morgan-Kaufmann, editor, *Machine learnig : an artificial intelligence approach*, pages 83–134. R. S. Michalski and J. G. Carbonell and T. M. Mitchell, 1983.

[Miet99] K. M. Miettinen. *Nonlinear multiobjective optimization*. 1999.

[Neld65] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7 :308–313, 1965.

[Noda99] E. Noda, A. A. Freitas, and H. S. Lopes. Discovering interesting prediction rules with a genetic algorithm. In *Proceedings of the Congress on Evolutionary Computation (CEC'99)*, pages 1322–1329. Washington D.C., USA, july 1999.

[Papa01] A. Papagelis and D. Kalles. Breeding decision trees using evolutionary techniques. In *Proceedings of the 18th International Conference on Machine Learning (ICML'01)*, pages 393–400. Morgan Kaufmann, San Francisco, CA, USA, 2001.

[Papp02] G. L. Pappa, A. A. Freitas, and C. A. A. Kaestner. A multiobjective genetic algorithm for attribute selection. In Nottingham Trent University, editor, *Proceedings of the 4th International Conference on Recent Advances in Soft Computing (RASC'02)*, pages 116–121. J. Garibaldi, A. Lofti and R. John, december 2002.

[Pei97] M. Pei, E. D. Goodman, and W. F. Punch. Pattern discovery from data using genetic algorithms. In *Proceedings of the 1st Pacific Asia Conference on Knowledge Discovery and Data Mining*. February 1997.

[Pent93] G. C. Pentzaropoulos and D. I. Giokas. Cost-performance modeling and optimization of network flow via linear goal programming analysis. *Computer Communications*, 16(10) :645–652, 1993.

[PS91] G. Piatetsky-Shapiro. Discovery, analysis and presentation of strong rules. In MIT Press, editor, *Proceedings of International Conference on Knowledge Discovery in Databases (KDD'91)*. G. Piatetsky-Shapiro and W. J. Frawley, 1991.

[Quin86] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1 :81–106, 1986.

[Quin91] J. R. Quinlan. Technical note : improved estimates for the accuracy of small disjuncts. *Machine Learning*, 6(1) :93–98, 1991.

[Quin96] J. R. Quinlan. Bagging, boosting and c4.5. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI'96)*, pages 725–730. Portland, Oregon, USA, august 1996.

[Radc95] N. J. Radcliffe. Ga-miner : parallel data mining with hierarchical genetic algorithms - final report. Technical report, 1995.

[Rear97a] B. J. Reardon. Fuzzy logic vs niched pareto multi-objective genetic algorithm optimisation : Part 1 : Schaffer's f2 problem. Technical report LA-UR-97-3675, Los Alamos National Laboratory, 1997.

[Rear97b] B. J. Reardon. Fuzzy logic vs niched pareto multi-objective genetic algorithm optimisation : Part 2 : a simplified born-mayer problem. Technical report LA-UR-97-3676, Los Alamos National Laboratory, 1997.

[Roy93] B. Roy and D. Bouyssou. *Aide multicritère à la décision : méthodes et cas*. 1993.

[Saka88] M. Sakawa and H. Yano. An interactive fuzzy satisficing method for multiobjective linear programming problems with fuzzy parameters. *Fuzzy Sets and Systems*, 28 :129–144, 1988.

[Saka02] M. Sakawa. *Genetic algorithms and fuzzy multiobjective optimization*. 2002.

[Scha85] J. D. Schaffer. Learning multiclass pattern discrimination. In *Proceedings of the International Conference on Genetic Algorithms and Their Applications (ICGA'85)*, pages 74–79. Pittsburgh, PA, 1985.

[Scha93] C. Schaffer. Overfitting avoidance as bias. *Machine Learning*, 10 :153–178, 1993.

[Seba88] M. Sebag and M. Schoenauer. Generation of rules with certainty and confidence factors from incomplete and incoherent learning bases. In *Proceedings of the European Conference on Knowledge Acquisition Workshop (EKAW'88)*. 1988.

[Shan49] C. E. Shannon and W. Weaver. *The mathematical theory of communication*. 1949.

[Siar94] P. Siarry. La méthode du recuit simulé en électronique. Rapport d'habilitation à diriger des recherches, Université de Paris-Sud, Orsay, Paris, France, 1994.

[Siar95] P. Siarry. Optimisation et classification de données - cours de dea gbm. Technical report, Université de Paris 12, Paris, France, april 1995.

[Srik96] R. Srikant and R. Agrawal. Mining sequential patterns : generalizations and performance impovements. In Springer-Verlag, editor, *Proceedings of the 5th Biennial International Conference on Extending Database Technology (EDMT'96) - Lecture notes in Computer Science*, pages 3–17. March 1996.

[Srin93] N. Srinivas and K. Deb. Multiobjective optimization using non-dominated sorting in genetic algorithms. Technical report, Department of Technical Engineering, Indian Institute of Technology, Kanput, Indian, 1993.

[Taba79] D. Tabak. Computer based experimentation with multicriteria optimisation problems. *IEEE Transactions on Systems Man and Cybernetics*, SMC9(10) :676–679, september 1979.

[Teus00] T. Teusan, G. Nachouki, H. Briand, and J. Philippe. Discovering association rules in large, dense databases. In Springer-Verlag, editor, *Proceedings of the 4th European Conference on Knowledge Discovery and Data Mining (PKDD'00) - Lecture notes in Artificial Intelligence 1910*, pages 638–645. Lyon, France, 2000.

[Thom99] J. D. Thomas and K. Sycara. Data mining with evolutionary algorithms : research directions. In AAAI Press, editor, *Papers from the AAAI'99/GECCO'99 Workshop - Technical report WS-99-06, 7-11*. A. A. Freitas, 1999.

[Thom00] J. D. Thomas and K. Sycara. In A. S. Wu, editor, *Proceedings of the Genetic and Evolutionary Computation Conference Workshop Program (GECCO'00) - Workshop on Data Mining with Evolutionary Algorithms*, pages 72–75. 2000.

[Ting94] K. M. Ting. The problem of small disjuncts : its remedy in decision trees. In *Proceedings of the 10th Canadian Conference on Artificial Intelligence (CCAI'94)*, pages 91–97. 1994.

[UCI] Irvine University UCI. Uci machine learning. ftp.ics.uci.edu/pub/machine-learning-databases.

[Veld99] D. A. Ven Veldhuizen. *Multiobjective evolutionary algorithms : classifications, analyses and new innovations.* Ph.D. thesis, Graduate School of Engineering - Air Force Institute of Technology, Wright Patterson AFB, Ohio, USA, january 1999.

[Vent96] G. Venturini, M. Slimane, F. Morin, and J.-P. Asselin de Beauville. On using interactive genetic algorithms for knowledge discovery in databases, 1996.

[Vinc89] Ph. Vincke. *L'aide multicritère à la décision.* 1989.

[Wang99] K. Wang, S. Zhou, and S. C. Liew. Building hierarchical classifiers using class proximity. In *Proceedings of the International Conference on Very Large Data Bases (VLDB'99)*, pages 363–374. Edinburgh, UK, september 1999.

[Weis95] G. M. Weiss. Learning with rare cases and small disjuncts. In *Proceedings of the 12th International Conference on Machine Learning (ICML'95)*, pages 558–565. Lake Tahoe, California, USA, 1995.

[Weis98] G. M. Weiss and H. Hirsh. The problem with noise and small disjuncts. In *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*, pages 574–578. Morgan Kaufmann Publishers, San Francisco, California, USA, 1998.

[Weis99] G. M. Weiss. Timeweaver : a genetic algorithm for identifying predictive patterns in sequences of events. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99)*, pages 718–725. Morgan Kaufmann, 1999.

[Weis00a] G. M. Weiss and H. Hirsh. A quantitative study of small disjuncts. In *Proceedings of the 17th National Conference on Artificial Intelligence*. Austin, Texas, USA, 2000.

[Weis00b] G. M. Weiss and H. Hirsh. A quantitative study of small disjuncts : experiments and results - expanded version of the paper. 2000.

[Weis01] G. M. Weiss and F. Provost. The effect of class distribution on classifier learning. Technical report ML-TR-43, Department of Computer Sciences, Rutgers University, january 2001.

[Zhan00] T. Zhang. Association rules. In Springer-Verlag, editor, *Proceedings of the PAKDD International Conference (PAKDD'00) - LNAI 1805*, pages 245–256. T. Terano and H. Liu and A. L. P. Chen, 2000.

[Zigh98] D. A. Zighed, S. Rabaseda, and R. Rakotomalala. Fusinter : a method for discretization of continuous attributes for supervised learning. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(33) :307–326, 1998.

[Zigh00] D. A. Zighed and R. Rakotomalala. *Graphes d'induction - Apprentissage et data mining*. 8, quai du Marché-Neuf, 75004 Paris, France, 2000.

Résumé

Les développements accrus des technologies numériques ont engendré depuis quelques années, des volumes de données extrêmement importants, qui peuvent receler des informations utiles pour les organismes qui les ont produites. Ce constat a donné naissance à un champ d'exploration : l'extraction de connaissances à partir des données, également appelée fouille de données, ou encore data mining dans la terminologie anglo-saxone. L'extraction de connaissances à partir des données désigne le processus non-trivial d'extraction d'informations implicites, précédemment inconnues et potentiellement utiles enfouies dans les données.

La fouille de données comprend globalement cinq phases qui sont la sélection des données, leur pré-traitement, leur transformation, l'extraction de modèles ou motifs et enfin l'évaluation de connaissances extraites. L'ensemble des travaux que nous présentons dans ce mémoire, s'inscrit dans ce contexte et en particulier dans la phase d'extraction de modèles proprement dite. Nous nous intéressons d'une part aux connaissances exprimées sous la forme de règles de dépendance et d'autre part à la qualité de ces règles. Une règle de dépendance représente une implication conditionnelle entre ensembles d'attributs du jeu de données et la qualité d'une règle est relative à sa précision, sa compréhensibilité, son utilité, et son intérêt pour l'utilisateur final. Les travaux existants relatifs aux mesures de qualité des règles de dépendance se basent généralement sur une étude analytique. Les algorithmes standards mis en œuvre ont pour but de rechercher les meilleurs modèles ; par exemple un algorithme d'induction d'arbres de décision génère le meilleur classifieur sur les instances traitées et ce en choisissant à chaque niveau de l'arbre, l'attribut le plus discriminant du jeu de données. En recherche de règles d'association, les algorithmes réalisent un parcours de l'espace de recherche afin de ne retenir que les modèles fournissant la meilleure description du jeu de données. Derrière ces processus se cache en fait une véritable problématique d'optimisation qui n'est pas explicitement exprimée.

La fouille de données considérée selon ce point de vue constitue l'essentiel de notre approche. Nous considérons la recherche des règles de dépendance les plus intéressantes comme étant un problème d'optimisation dans lequel la qualité d'une règle est quantifié par le biais d'une ou de plusieurs mesures. Il ressort que la plupart des mesures observées présentent des propriétés différentes suivant le jeu de données et de fait, une approche analytique est difficilement envisageable sans fixer certains paramètres. Ainsi, étant donné que nous considérons la fouille de données sous l'angle de l'optimisation, il est nécessaire d'étudier les espaces de recherche induits par les mesures ainsi que les algorithmes de recherche dans ces espaces. Nous observons les variations relatives de mesures évaluées simultanément ; certaines d'entre elles sont antagonistes ce qui ne permet pas d'obtenir « la » meilleure règle ; il faut alors considérer un ensemble de compromis satisfaisants. Nous présentons dans ce mémoire, plusieurs approches permettant de mettre en évidence des particularités des espaces de recherche et nous apportons des solutions par le biais de l'approche évolutionnaire. Notre objectif est donc d'extraire des modèles à base de règles de dépendance en mettant en œuvre des algorithmes d'extraction permettant l'optimisation d'une mesure ou d'un ensemble de mesures de qualité. Nous nous sommes orientés vers une approche évolutionnaire et en particulier vers les algorithmes génétiques, métaheuristique éprouvée et très efficace sur de vastes espaces de recherche comme ils se présentent en fouille de données. D'autre part, ces algorithmes permettent de mettre en œuvre l'optimisation multi-objectifs et donc de prendre en compte plusieurs mesures de qualité et d'extraire les modèles représentant les meilleurs compromis vis à vis des critères choisis par l'utilisateur.

Mots clés : extraction de connaissances à partir des données, base de données, algorithme évolutionnaire, règle de dépendance, mesure de qualité, optimisation multi-objectifs.