Mémoire de thèse présenté par

Mohammed KHABZAOUI

pour obtenir le titre de

Docteur en Informatique

# Modélisation et résolution multi-objectifs des règles d'association : Application à l'analyse de données biopuces

Thèse soutenue le 20 novembre 2006 devant la Commission d'Examen

| | | |
|---|---|---|
| Président : | Bernard Toursel | Professeur, LIFL - USTL |
| Rapporteurs : | Jin-Kao Hao | Professeur, Université de Angers |
| | Jean-Marc Pierson | Professeur, Université Paul Sabatier, Toulouse |
| Examinateurs : | Gilles Goncalves | Professeur, Faculté des Sciences Appliquées - LGI2A, Béthune |
| Directeurs : | Clarisse Dhaenens | Professeur, LIFL - USTL |
| | El-Ghazali Talbi | Professeur, LIFL - USTL |

Université des Sciences et Technologies de Lille
LIFL - UMR 8022 - Cité Scientifique, Bât. M3 - 59655 Villeneuve d'Ascq Cedex

A la mémoire de mon père et mon frère,
à Samia et notre fils Kaïs,
à ma mère, mes frères et mes sœurs,
et à la mémoire de Sébastien CAHON.

# Remerciements

Je tiens à exprimer tous mes remerciements et ma profonde gratitude à M$^{me}$ C. DHAENENS et M. EG-TALBI pour l'honneur qu'ils m'ont fait en me proposant et dirigeant cette thèse, et la confiance exprimée à mon égard dans le travail mené.

Je remercie M. Bernard TOURSEL pour m'avoir fait l'honneur de présider mon jury de ma thèse.

Je remercie mes rapporteurs, Jin-Kao HAO et M. Jean-Marc PIERSON, pour leurs commentaires avertis sur mes travaux de recherche.

Je remercie M. Gilles GONCALVES de m'avoir fait l'honneur d'examiner le travail de ma thèse.

Je remercie les membres de l'équipe OPAC pour tous les échanges et discussions constructifs que nous avons eu depuis mon arrivée et pour la bonne ambiance de travail qui règne en sein de l'équipe.

Je tiens enfin à remercier vivement Samia et notre fils Kaïs d'avoir compris et accepté ma faible disponibilité pendant la préparation de ma thèse. Merci à Samia pour son précieux soutien et ses encouragements.

# Table des matières


**1 Introduction générale** **7**

**2 Les Puces à ADN** **11**

2.1 Introduction . . . . . 11

2.2 Principe des puces à ADN . . . . . 12

2.2.1 Structure de l'ADN . . . . . 12

2.2.2 Dépot direct d'ADNc . . . . . 14

2.2.2.1 Fonctionnement . . . . . 14

2.2.2.2 Acquisition des données d'expression . . . . . 15

2.2.2.3 Normalisation des données . . . . . 16

2.2.3 Technologie Affymetrix . . . . . 16

2.2.3.1 Fonctionnement . . . . . 16

2.2.3.2 Traitement de données et logiciel MAS5 . . . . . 17

2.3 Banques de données . . . . . 20

2.3.1 Banques de données d'expression de gènes . . . . . 20

2.3.2 Gene Ontology (GO) . . . . . 22

2.4 Applications . . . . . 23

2.4.1 Analyses d'expression de gènes . . . . . 24

2.4.2 Action des médicaments . . . . . 24

2.4.3 Analyses d'ADN génomique . . . . . 24

2.5 Différentes techniques de datamining pour les puces à ADN . . . . . 25

2.5.1 Classification supervisée . . . . . 26

2.5.2 Classification non supervisée . . . . . 27

2.6 Conclusion . . . . . 28

**3 Modélisation multi-objectif des règles d'association** **31**

3.1 Introduction . . . . . 31

3.2 Datamining . . . . . 32

3.3 Règle d'association . . . . . 34

3.3.1 Problématique . . . . . 34

3.3.2 Algorithmes pour les règles d'association . . . . . 36

3.3.2.1 Algorithme Apriori . . . . . 37

3.3.2.2 Autres algorithmes . . . . . 37

3.3.2.3 Versions parallèles . . . . . 40

3.4 Mesures de qualité existantes . . . . . 40

3.5 Propriétés des bonnes mesures . . . . . 46

3.5.1 Première approche : Propriétés probabilistes . . . . . 46

3.5.2 Approche fonctionnelle . . . . . 48

3.5.3 Troisième approche : Préférences utilisateur (expert) . . . . . 49

3.6 Notre approche : Analyse statistique des critères . . . . . 50

3.6.1 Analyse descriptive univariée . . . . . 51

3.6.2 Analyse descriptive multivariée : ACP . . . . . 51

3.7 Conclusion . . . . . 59

**4 L'optimisation combinatoire multi-objectif** **61**

4.1 Introduction . . . . . 61

4.1.1 Optimisation combinatoire . . . . . 62

4.1.2 Problème d'optimisation multi-objectif . . . . . 62

4.2 Définitions et vocabulaire . . . . . 63

4.2.1 Relations d'ordre et de dominance . . . . . 63

4.2.2 Front Pareto . . . . . 63

4.3 Algorithmes de résolution . . . . . 64

4.3.1 Méthodes de recherche locale . . . . . 65

4.3.1.1 Descente locale . . . . . 66

4.3.1.2 Recuit simulé . . . . . 66

4.3.1.3 Algorithme glouton aléatoire (GRASP) . . . . . 68

4.3.1.4 Recherche Tabou . . . . . 69

4.3.2 Métaheuristiques à population de solutions . . . . . 70

4.3.2.1 Algorithmes génétiques . . . . . 70
4.3.2.2 Recherche dispersée . . . . . 71
4.3.2.3 Colonies de fourmis . . . . . 71
4.4 Classification des approches . . . . . 72
4.4.1 Méthodes scalaires . . . . . 73
4.4.2 Approches non-Pareto et non-scalaires . . . . . 75
4.4.3 Approches Pareto . . . . . 76
4.4.3.1 Méthodes de ranking . . . . . 76
4.4.3.2 Elitisme . . . . . 78
4.4.3.3 Mécanisme de diversité . . . . . 78
4.5 Evaluation de performances en optimisation multi-objectif . . . . . 79
4.5.1 Les métriques absolues sans référence . . . . . 79
4.5.2 Les métriques absolues avec une référence . . . . . 80
4.5.3 Les métriques relatives . . . . . 82
4.5.4 Mesures utilisées . . . . . 84
4.5.5 Guimoo : Une interface graphique pour les problèmes d'optimisation multi-objectifs . . . . . 84
4.6 Conclusion . . . . . 86

**5 Algorithmes génétiques pour les règles d'association multi-objectifs 89**
5.1 Algorithme génétique pour les règles d'association . . . . . 89
5.1.1 Codage et représentation des solutions . . . . . 90
5.1.2 Génération de la population initiale . . . . . 90
5.1.3 L'opérateur de croisement . . . . . 92
5.1.3.1 Croisement par mutation de valeurs . . . . . 92
5.1.3.2 Croisement par insertion d'attributs . . . . . 92
5.1.4 L'opérateur de mutation . . . . . 93
5.1.5 La mutation adaptative . . . . . 94
5.2 Mécanismes et opérateurs multi-objectifs . . . . . 94
5.2.1 La sélection . . . . . 95
5.2.2 Elitisme et archive Pareto . . . . . 96
5.2.2.1 La sélection élitiste . . . . . 96
5.2.2.2 Remplacement élitiste . . . . . 96

5.3 Implémentation . . . . . 97
5.3.1 EO : plateforme de développement . . . . . 97
5.3.2 MOEO : Optimisation Multi-objectif avec EO . . . . . 99
5.3.2.1 Elitisme . . . . . 101
5.3.2.2 Opérateurs de diversification . . . . . 102
5.3.2.3 Opérateurs adaptatifs . . . . . 102
5.4 Résultats expérimentaux . . . . . 103
5.4.1 Bases de données . . . . . 103
5.4.2 Analyse des opérateurs . . . . . 104
5.4.2.1 Stabilité de l'algorithme . . . . . 105
5.4.2.2 Adaptativité des mutations . . . . . 105
5.4.2.3 L'élitisme . . . . . 106
5.4.2.4 Projections 2D . . . . . 106
5.4.3 Résultats : Règles d'association . . . . . 107
5.5 Conclusion . . . . . 107

**6 Méthodes coopératives pour les règles d'association multi-objectif 113**
6.1 Introduction . . . . . 113
6.2 Approche parallèle . . . . . 114
6.2.1 Modèles parallèles pour les AGs . . . . . 114
6.2.1.1 Modèle centralisé : l'évaluation parallèle . . . . . 114
6.2.1.2 Modèle cellulaire : population distribuée . . . . . 114
6.2.1.3 Modèle insulaire : Modèle en îles . . . . . 115
6.2.2 Algorithme génétique parallèle proposé . . . . . 116
6.2.3 Politique d'échange . . . . . 116
6.2.4 PARADISEO : plateforme de développement . . . . . 121
6.2.4.1 Le modèle en îles . . . . . 122
6.2.4.2 La parallélisation de la fonction objectif . . . . . 123
6.2.4.3 La parallélisation de la phase d'évaluation . . . . . 124
6.2.4.4 Règles d'association avec ParadisEO . . . . . 124
6.2.5 Validation du modèle parallèle . . . . . 125
6.3 Approche hybride . . . . . 127
6.3.1 Méthode exacte (Procédure énumérative) . . . . . 128

6.3.2 Schéma d'hybridation AG/exacte . . . . . 130
6.3.3 Evaluation du modèle . . . . . 131
6.4 Applications : intégration du module . . . . . 133
6.4.1 Projet GGM : vers une grille biomédicale . . . . . 133
6.4.2 BASE et le plugin Rule mining . . . . . 134
6.4.3 Schéma de BASE . . . . . 136
6.4.3.1 Le plugin Rule mining . . . . . 138
6.4.3.1.1 Visualisation 3D . . . . . 138
6.4.3.1.2 Visualisation de tous les critères . . . . . 140
6.4.3.1.3 Enrichissement des règles avec GO (Gene Ontology) . . . . . 140
6.5 Conclusion . . . . . 142

**7 Conclusion générale et perspectives** **143**

# Chapitre 1

# Introduction générale

Cette thèse s'inscrit dans le cadre des travaux de recherche en optimisation combinatoire menés par l'équipe OPAC (Optimisation PArallèle et Coopérative) au sein du Laboratoire d'Informatique Fondamentale de Lille (LIFL) et du projet INRIA DOLPHIN de l'UR Futurs. Le travail réalisé porte sur l'extraction de connaissances à partir des puces à ADN. L'approche choisie consiste à rechercher des règles d'association à l'aide d'algorithmes génétiques.

Les puces à ADN permettent de visualiser simultanément le niveau d'expression de plusieurs milliers de gènes ou groupe de gènes dans des conditions différentes (physiologiques ou pathologiques). Cette technologie haut débit génère une grande diversité de données qui implique un important travail d'analyse. Un grand nombre de techniques liées à l'informatique sont nécessaires à l'analyse des données issues de cette technologie : analyse d'images, stockage et gestion des informations, techniques de normalisation, analyses statistiques, représentations graphiques. Aux vues des dimensions et des quantités de données, il nous paraît alors évident que les techniques de datamining, d'apprentissage et de statistique sont incontournables.

Dans notre travail, nous nous focalisons sur la phase de datamining. Aujourd'hui, de nombreuses méthodes de classification et segmentation (classification hiérarchique, k-moyennes, SOM, KNN, SVM...) se sont montrées particulièrement efficaces pour regrouper et classer les gènes. Néanmoins, ces méthodes ne permettent en général de découvrir qu'une partie des relations parmi toutes les relations potentielles entre les gènes. De plus, la plupart du temps, les classes recherchées doivent être vérifiées sur l'ensemble des expérimentations et un gène ne peut appartenir qu'à une seule classe. C'est pourquoi nous proposons d'utiliser une approche plus générale : les règles d'association. En effet, cette méthode permet de mettre en évidence des relations plus précises entre les gènes.

La recherche des règles d'association consiste à extraire un ensemble de formules logiques conditionnelles qui déduisent la valeur d'un attribut but à partir des valeurs d'autres attributs apparaissant simultanément. La combinatoire du choix des attributs à faire figurer dans la règle et leurs différentes valeurs possibles, font que la taille de l'espace de recherche des règles candidates est exponentielle. Anguilli [AIP01] a montré que le problème de recherche de règles d'association se réduit en un problème de *clique* qui a déja

été démontré NP-complet [GJ79].

Afin de traiter le problème le problème de recherche de règles d'association comme un problème d'optimisation combinatoire, des questions fondamentales sont : qu'est-ce qu'une règle pertinente ? Comment évaluer la qualité d'une règle ? La qualité d'une règle peut dépendre de plusieurs caractéristiques (sa force de prédiction, son nombre d'occurrences ...) en fonction du contexte. Cela a donné lieu à la définition d'un grand nombre de critères mesurant cette qualité. En effet, plusieurs mesures de qualité des règles ont été proposées par différentes communautés telles que la statistique, le datamining ou l'optimisation. Malheureusement, il n'existe pas de mesure universelle, reconnue par tous [TKS02, HH99, PS91].

Nous avons alors réalisé une analyse statistique afin de mettre en évidence les corrélations existantes entre les mesures. Cette analyse permet de regrouper les critères ayant le même comportement (et mesurant donc les mêmes propriétés) et de déterminer un ensemble cohérent de critères complémentaires [KDNT03, KDT06a].

On peut alors formuler le problème de recherche de règles d'association sous la forme d'un problème d'optimisation multi-objectif, où l'on cherche à optimiser plusieurs critères complémentaires [KDT04a, KDT04b]. C'est cette approche que nous avons voulu étudier dans cette thèse.

Les méthodes de résolution des problèmes d'optimisation multi-objectifs peuvent être classées en deux grandes classes : les méthodes exactes et les méthodes approchées (heuristiques et métaheuristiques)[DK01, CS02].

Les méthodes exactes examinent, souvent de manière implicite, la totalité de l'espace de recherche. Ainsi, elles ont l'avantage de produire une solution optimale lorsqu'aucune contrainte de temps n'est donnée. Néanmoins, le temps de calcul nécessaire pour atteindre une solution optimale peut devenir vite prohibitif, ce malgré les diverses techniques et heuristiques qui ont été développées pour accélérer l'énumération des solutions. Dans de telles situations (et dans beaucoup d'autres), les méthodes approchées constituent une alternative indispensable et complémentaire. Le but d'une telle méthode n'est plus de fournir une solution optimale au problème donné. Elle cherche avant tout à produire une solution sous-optimale de meilleure qualité possible avec un temps de calcul raisonnable.

Dans notre étude, étant donné le nombre de combinaisons possibles pour la construction de la règle, la combinatoire associée au problème est très importante. Ceci ne permet pas d'utiliser pour des problèmes de grandes tailles (comme c'est le cas ici) des algorithmes exacts d'énumération. Il est donc nécessaire d'avoir recours à des heuristiques ou des métaheuristiques telles que les algorithmes génétiques. En particulier, ces dernières années, les méthodes basées sur les algorithmes génétiques ont connu un succès grandissant dans les domaines de la recherche opérationnelle et de l'intelligence artificielle, où elles sont utilisées pour résoudre des problèmes d'optimisation combinatoire multi-objectifs et d'apprentissage.

Le contexte de la thèse étant la recherche de règles d'association en utilisant une approche multi-objectif, nous nous focalisons dans ce mémoire sur la modélisation et la résolution multi-objectif de ce problème. Ce problème étant fortement combinatoire, nous nous intéressons plus particulièrement à l'apport des méthodes d'optimisation approchées,

à savoir les algorithmes génétiques (AG) pour lesquels nous proposons des opérateurs spécifiques. En vue de meilleures performances, nous étudions la combinaison de méthodes permettant cette recherche de règle d'association multi-objectifs. Nous nous intéressons d'une part à l'hybridation entre une méthode exacte et une méthode approchée [KDT06b]. D'autre part, nous nous penchons plus particulièrement sur des méthodes parallèles et coopératives entre méta-heuristiques et proposons un modèle en îles où plusieurs AG sont déployés pour faire évoluer et coopérer simultanément différentes populations (îles) de solutions [KDT05a, KDT05b]. Pour l'implémentation de nos algorithmes, nous avons choisi d'utiliser une plateforme de développement pour les métaheuristiques, intègrant des composants logiciels pour l'optimisation combinatoire multiobjectif en utilisant des architectures du calcul parallèle et distribué sur une grappe de machines ou une grille dédiée. Cette plateforme, ParadisEO (PARAllel and DIStributed Evolving Objects), a été développée dans l'équipe [Cah05].

Dans le but de faciliter l'analyse de données obtenues à l'aide de la technologie des puces à ADN, nous avons inséré un module appelé Rulemining dans la plate forme Bio-Array Software Environment (BASE) [STVC+02] qui est une base de données permettant de gérer et de stocker d'importante quantité de données générées par des analyses de biopuces. BASE gère les informations biologiques, les données brutes et les images. Il possède également des outils de normalisation, de visualisation et d'analyse des données.

Le reste du mémoire s'articule en 5 chapitres.

Le **chapitre 2** est ainsi consacré aux puces à ADN. Nous présenterons en premier lieu la structure de l'ADN et le principe des puces à ADN ainsi que les différentes technologies utilisées pour leur fabrication. Ensuite, nous évoquerons les différentes techniques de datamining utilisées pour les puces à ADN.

Le **chapitre 3** est dédié aux règles d'association en datamining. Nous présenterons dans un premier temps le data mining et ses différentes techniques. Ainsi, nous présenterons le problème de la recherche de règles d'association et les méthodes de résolution de la littérature.

Dans la deuxième partie du chapitre, nous aborderons en premier lieu les différents indicateurs mesurant la qualité des règles d'association. Puis, nous évoquerons les propriétés et préférences proposées par différents auteurs et nous terminerons par notre étude statistique de ces critères en vue de déterminer les critères à utiliser lors d'une approche par optimisation combinatoire multi-objectif.

Le **chapitre 4** dresse un état de l'art non exhaustif des domaines de l'optimisation multi-objectif et des métaheuristiques. Dans un premier temps, nous présenterons le contexte de l'optimisation combinatoire multi-objectif, nous introduisons des concepts fondamentaux tels que la dominance, la surface de compromis, Pareto, ranking. Puis, nous présenterons les principales méthodes de résolution (les méthodes à base de recherche locale et les approches évolutionnaires). Nous terminerons le chapitre par une présentation de quelques mesures permettant d'évaluer la qualité des solutions Pareto produites par un algorithme.

Le **chapitre 5** présentera notre travail sur les règles d'associations. Nous proposerons

une résolution multiobjectif pour les règles d'association en utilisant les algorithmes génétiques. Nous présenterons l'algorithme génétique que nous proposons en détaillant les opérateurs et les mécanismes adaptatifs et multi-objectifs que nous avons utilisés. Nous validerons notre approche sur des bases de données classiques de data mining et sur des données biopuce (puces à ADN).

Le **chapitre 6** concerne l'approche coopérative et hybride que nos proposons. Nous présenterons dans un premier temps une approche coopérative parallèle développée pour le problème de recherche de règles, dans laquelle différents algorithmes génétiques coopèrent. Dans un deuxième temps nous présenterons une approche coopérative entre une méta-heuristique et un algorithme énumératif. Enfin nous présenterons deux applications dans lesquelles l'algorithme a été intégré.

Nous terminerons ce mémoire par différentes conclusions, ainsi que des perspectives de recherche qui nous semblent intéressantes pour continuer ce travail.

# Chapitre 2

# Les Puces à ADN

Sommaire


2.1 Introduction . . . . . 11
2.2 Principe des puces à ADN . . . . . 12
2.3 Banques de données . . . . . 20
2.4 Applications . . . . . 23
2.5 Différentes techniques de datamining pour les puces à ADN . 25
2.6 Conclusion . . . . . 28


## 2.1 Introduction

Du fait du potentiel qu'on leur attribue pour le diagnostic biologique et en génomique, les puces à ADN suscitent un intérêt croissant de la part des scientifiques et des industriels. Les puces à ADN, que l'on appelle aussi "biochips" ou "genechips" en anglais, sont apparues au milieu des années 90. Elles sont au confluent de la micro-électronique, de la chimie combinatoire, de la biologie moléculaire, de l'informatique et du traitement du signal. Ces petits carrés de verre, de polymères ou parfois de silicium, sont capables de distinguer en une seule opération et par un traitement massivement parallèle quelques dizaines de milliers de séquences d'acides nucléiques. Elle permettent de mesurer et de visualiser très rapidement les différences d'expression entre les gènes. Il est possible de repérer des mutations et de savoir quels gènes répondent à l'action d'une molécule ou sont impliqués dans une maladie. Elles sont appliquées au diagnostic médical et aux recherches thérapeutiques [BB99].

La technologie haut débit des puces à ADN nécessite d'organiser et d'analyser de très grandes quantités de données. On trouve ainsi des informations sur les échantillons hybridés, sur les images de puces et les matrices de valeurs qui en découlent, sur la conception des puces elle-même, ou encore sur les molécules employées. BioArray Software Environment (BASE) est une plate-forme web spécialisée dans la gestion, le stockage et l'analyse de données des expériences de puces à ADN

Dans le but de faciliter l'analyse de données obtenues à l'aide de la technologie des puces à ADN, nous avons réalisé un module de règles d'association en data mining pour les données issues de puces à ADN dans la plate forme BASE basé sur les algorithmes évolutionnaires (voir chapitres 5 et 6) .

Dans ce chapitre, nous présenterons les puces à ADN. Nous rappelons en premier lieu la structure de l'ADN, le principe de base des puces à ADN ainsi que leurs technologies. Ensuite, nous présenterons quelques applications majeures de cette technologie. Nous terminerons ce chapitre par différentes techniques d'analyse de données pour les puces à ADN.

## 2.2 Principe des puces à ADN

Depuis une vingtaine d'années, plusieurs techniques de biologie moléculaire ont été développées afin d'étudier le transcriptome (ARNm). Les premières approches proposées, le *Southern blot* et le *Northern blot*, permettent d'identifier et localiser une séquence particulière (sonde d'ARNm ou ADNc) dans un génome entier (cible) ou tout autre mélange complexe d'ADN. Ces techniques se limitent à l'analyse d'un petit nombre de gènes à la fois et ne permettent pas d'appréhender la complexité du phénomène de la transcription. Plus récemment, la technique SAGE (Serial Analysis of Genes Expression), permet d'identifier et quantifier, simultanément, le niveau d'expression de plusieurs milliers de gènes, dans un type cellulaire donné [VZVK95]. Cette méthode consiste à réaliser un inventaire des transcrits par séquençage en série de courts fragments d'ADNc (9 à 14 pb) ou *sequence tags*. Cette méthode est très sensible mais aussi très longue à mettre en œuvre, coûteuse et se limite à l'évaluation des niveaux d'expression des gènes. Parallèlement à la méthode SAGE, s'est développée la technologie des puces à ADN [SSDB95, LDB+], moins coûteuse et surtout plus évolutive en terme d'applications. En effet, les puces à ADN permettent non seulement de visualiser, simultanément, le niveau d'expression de plusieurs milliers de gènes dans un type cellulaire et un contexte physiologique et/ou pathologique particulier, mais aussi d'étudier la séquence des gènes dans un échantillon, les mutations ou le polymorphisme [MBdSD04]. Elles sont donc rapidement devenues un outil privilégié pour l'analyse du transcriptome. Pour comprendre le fonctionnement des puces à ADN, un petit rappel sur la structure de l'ADN est nécessaire.

### 2.2.1 Structure de l'ADN

**Définition :** L'ADN (Acide DésoxyriboNucléique) est la forme de stockage de l'information génétique du génome. Cette information est représentée sur le chromosome par une suite linéaire de gènes séparés par des régions intergéniques.

L'ADN est un acide nucléique formé par la répétition de sous-unités appelées *nucléotides*. Les nucléotides sont constitués de trois éléments : l'acide phosphorique + un sucre + une base azotée. Les bases azotées sont au nombre de quatre : Adénine, Guanine, Thymine et la Cytosine. L'ADN est composé de deux brins associés l'un à l'autre et enroulés sous forme d'hélice, dite structure secondaire, qui est le résultat de l'appariement des bases azo-

tées de façon complémentaire, l'Adénine avec la Thymine et la Cytosine avec la Guanine. Les paires de base de l'ADN sont perpendiculaires à l'axe de la double hélice.

Chaque organisme a un code unique qui contrôle son développement et son fonctionnement. Le nombre et l'arrangement de ces bases déterminent qui nous sommes, notre apparence et les maladies auxquelles nous sommes prédisposés. La structure schématisée de l'ADN est donnée sur la figure 2.1.

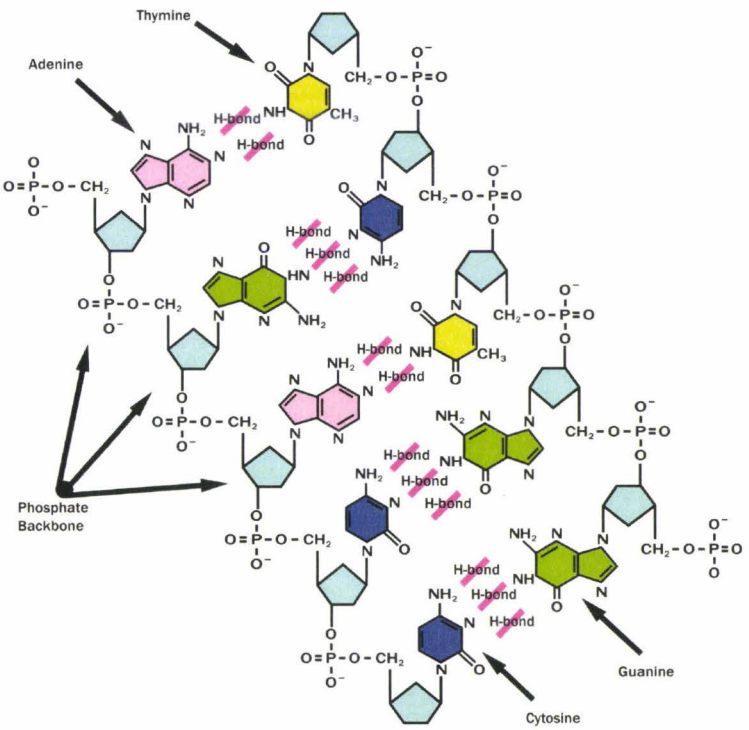



FIG. 2.1 – L'hybridation de la molécule d'ADN.

L'ADN a la propriété remarquable de passer d'une structure simple brin à une structure double brin de manière réversible ; c'est cette réaction d'association mieux connue sous le nom de réaction d'hybridation, utilisée en biologie moléculaire pour l'étude de l'ADN dans des conditions physico-chimiques données, qui sera la base du fonctionnement de la puce ADN.

En effet, les brins extraits de la double hélice d'ADN ont la capacité de reformer spontanément cette double hélice dès qu'ils se retrouvent face au brin complémentaire.

Le jumelage de plusieurs technologies a permis la miniaturisation de ces techniques d'hybridation, permettant ainsi de déceler des milliers de molécules d'acide nucléique de façon simultanée sur des matrices solides mesurant quelques centimètres carrés. Deux procédés majeurs de fabrication de puces à ADN sont couramment utilisés : (a) le dépot direct d'ADNc sur lamelle de verre activée, ou (b) la synthèse *in situ* d'oligonucléotides par photolithographie (technologie Affymetrix).

### 2.2.2 Dépot direct d'ADNc

#### 2.2.2.1 Fonctionnement

Le premier type de puce à ADN consiste en une lamelle de verre (identique à celle utilisée en microscopie traditionnelle) sur laquelle des milliers d'ADNc sont déposés à l'aide d'un micropipetteur robotisé. Grâce à cette technique, chacun des gènes (de fonction connue ou inconnue) est représenté par un seul point sur la lamelle. En général, deux échantillons d'ARN (sous forme d'ADNc obtenus par transcription inverse) sont co-hybridés sur la puce à ADNc. Les deux échantillons marqués par un fluorochrome différent (Cy-3 vert ou Cy-5 rouge) s'hybrident simultanément avec les molécules complémentaires sur la puce. La puce est lue par un scanner afin de mesurer l'intensité du signal lumineux mesurée aux deux longueurs d'ondes correspondant aux différents fluorochromes. Le rapport de fluorescence rouge/vert est ainsi déterminé. Il permet de comparer les taux d'expression relatifs de chacun des gènes pour les deux échantillons d'ADNc. Un excès du gène X dans l'échantillon marqué en rouge produira un signal rouge au point représentant le gène, un excès du gène Y dans l'échantillon marqué en vert produira un signal vert ; enfin, une expression équivalente du gène Z dans les deux échantillons produira un signal jaune (voir figure 2.2).

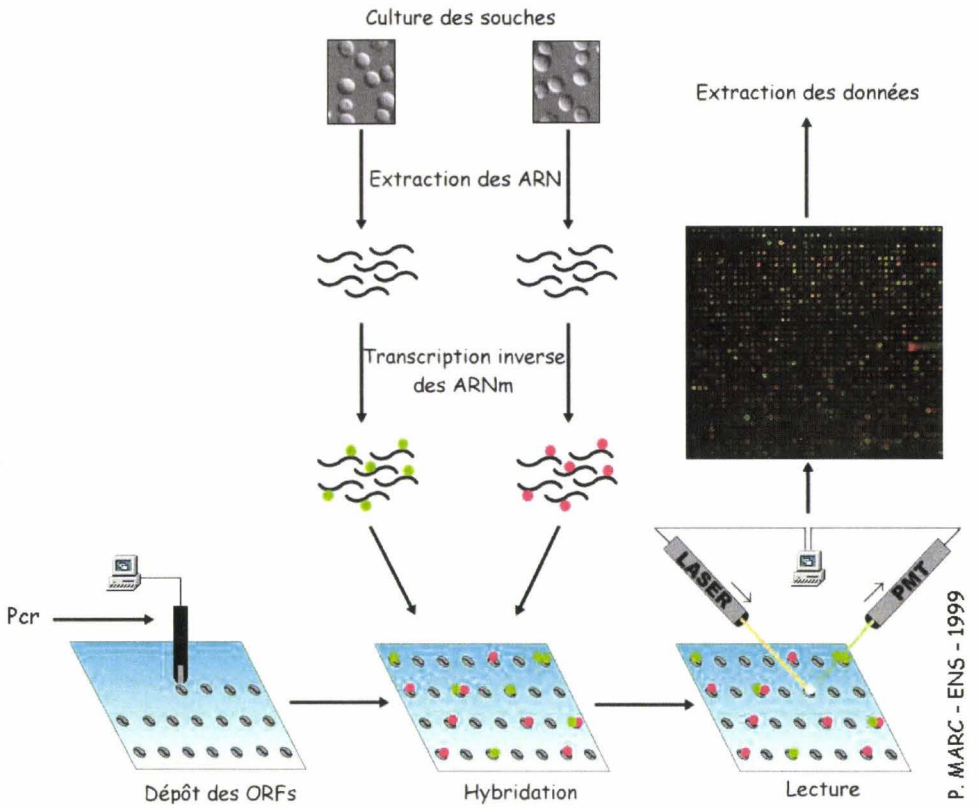



FIG. 2.2 – Dépot direct d'ADNc sur lamelle de verre (deux fluorochromes).

#### 2.2.2.2 Acquisition des données d'expression

Suite à la lecture des puces à ADN par un scanner, les niveaux d'expression sont estimés grâce à des logiciels d'analyse d'images (par exemple : Genepix Pro (Axon software), ScanAlyze (M. Eisen, Stanford University)). Ces logiciels extraient des informations qualitatives et semi-quantitatives pour chaque spot dans chacun des fluorochromes. Le but est de convertir l'image en valeurs numériques quantifiant l'expression des gènes. Le traitement des images est un aspect clé de l'extraction des données de puces à ADN. L'interprétation biologique des données, comme le nombre de gènes reporters détectés, dépend en partie de la qualité des logiciels d'analyse d'images. Globalement, les logiciels d'analyse d'images sont basés sur le même principe et possèdent la même procédure de traitement qui se déroule en trois étapes (figure 2.3) :

- Localisation des spots sur la lame. Cela permet de préciser les coordonnées de chaque spot sur l'image.
- Délimitation des pixels correspondants à la zone d'hybridation et classement des pixels en tant que signal ou bruit de font.
- Calcul de l'intensité globale de fluorescence pour chaque spot.

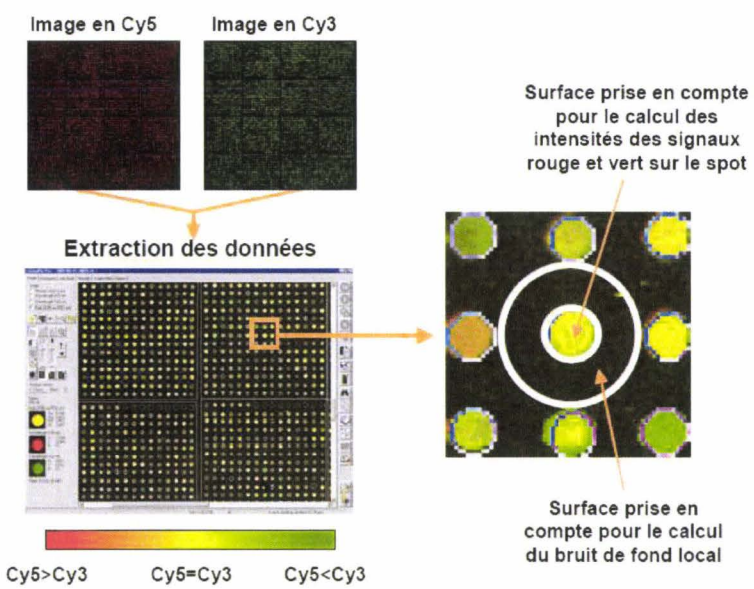



FIG. 2.3 – Analyse d'image.

Le niveau d'expression génique correspond à une mesure relative des intensités de fluorescence en Cy5 et Cy3. Un filtrage des données est réalisé pour sélectionner les gènes présentant une variation d'expression significative. Le tri des spots est basé sur une valeur seuil définie pour plusieurs critères de qualité. Les spots obtiennent un bon score s'ils possèdent des valeurs supérieures aux seuils. Les intensités des signaux, les bruits de fond et le nombre de pixels d'un spot dont l'intensité est au moins supérieure à une ou deux fois le bruit de fond moyen sont, en règle générale, les critères considérés. Les gènes répondant à

ces critères de qualité sont alors analysés pour définir le ratio d'induction ou de répression, c'est-à-dire ceux dont le ratio est significativement différent de 1. En effet, on établit pour chaque gène le rapport : fluorescence essai/fluorescence témoin : si ce rapport est inférieur à 1, le gène est dit réprimé (on parle de ratio de répression) ; si ce rapport est égal à 1, le gène est dit invariant ; si ce rapport est supérieur à 1, le gène est dit surexprimé ou induit (on parle alors de ratio d'induction).

#### 2.2.2.3 Normalisation des données

Pour pouvoir comparer des données entre plusieurs expériences, il est essentiel de les normaliser. Il convient donc de corriger, si nécessaire, les intensités en Cy3 et Cy5 pour éliminer les artefacts dus au protocole expérimental, comme par exemple : la qualité des lames, la quantité d'ARN, la différence de marquage des ARN avec les fluorochromes. La plupart des logiciels d'analyse normalisent à partir de la médiane ou de la moyenne des intensités (la normalisation par rapport à la moyenne globale des intensités, la normalisation par rapport à des spots témoins, la normalisation de Lowess etc).

Deux normalisations sont possibles :
- Normalisation intra-puce : régression linéaire, exponentielle ou approches par régressions non paramétriques [YDL+02] (en particulier l'algorithme de lowess fitness (LOcally WEighted Scatter plot Smoothing) et ses variantes sont actuellement les plus employées).
- Normalisation inter-puces : il est parfois nécessaire d'appliquer une normalisation inter-puces (scaling) afin de réduire l'écart de la variance des mesures entre les puces (médiane).

### 2.2.3 Technologie Affymetrix

#### 2.2.3.1 Fonctionnement

Le second type de puce à ADN, proposé par la société Affymetrix, est constitué d'oligonucléotides synthétisés directement sur un substrat solide par photolithographie. La synthèse d'un oligonucléotide de 25 paires de bases occupe un carré de 20 $\mu$m x 20 $\mu$m. La surface d'une puce est d'environ 1,28 $cm^2$, et peut contenir 400 000 oligonucléotides différents ! Une puce à ADN destinée à des études d'expression contient pour chaque gène un ensemble d'oligonucléotides mimant la séquence du gène, souvent choisi dans sa région 3', réduisant ainsi les risques d'hybridations croisées avec des séquences homologues de ce gène. Des oligonucléotides, dont la séquence varie pour une seule base, sont également ajoutés, ce qui permet de confirmer que le signal obtenu pour chacun des gènes est bien spécifique. Chaque oligonucléotide possède son propre contrôle d'hybridation. La concentration de l'ARN est mesurée par la moyenne des différences des oligonucléotides *Sonde* et *Cible*.

Affymetrix a opté pour la synthèse *on chip* (in situ) où les quatre bases constitutives de l'ADN sont déposées successivement dans l'ordre qui caractérise la sonde sur le support de verre. Cette opération est réalisée par un procédé de photo-déprotection localisée grâce

à un jeu de masques (figure 2.4). L'utilisation des masques permet de contrôler l'endroit et l'ordre d'addition des nucléotides.

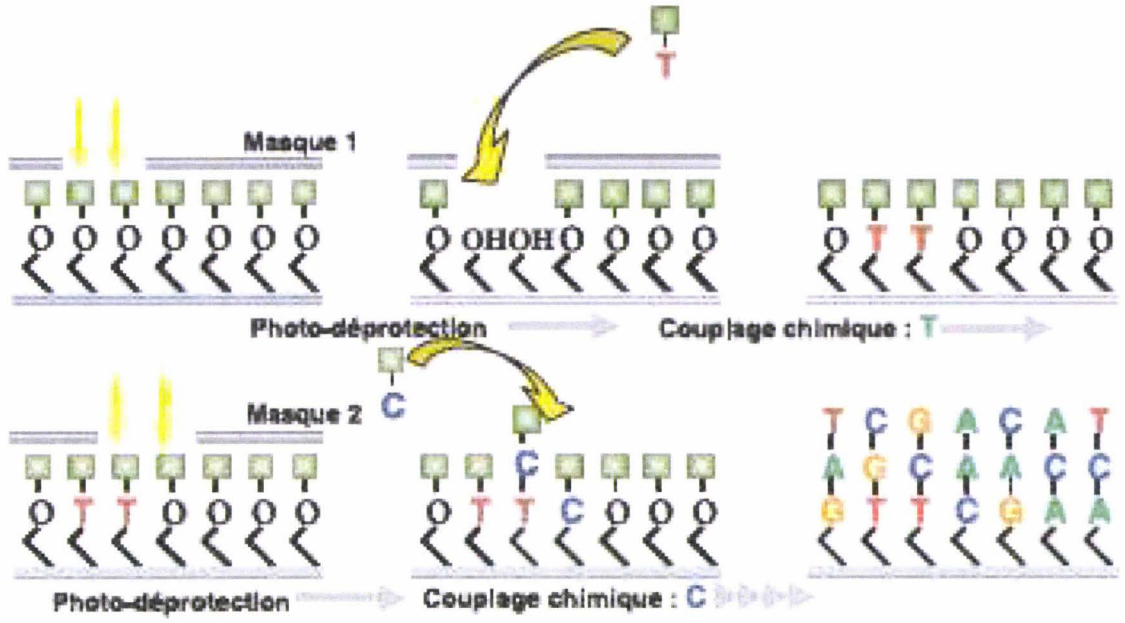



FIG. 2.4 – Procédé de photo-déprotection.

Une puce Affymetrix produit un signal d'une intensité donnée pour chaque groupe de sondes correspondant à un même gène. Chaque groupe de sondes est composé de 11 ou de 16 paires de sondes. Chaque paire consiste en une sonde ayant une correspondance parfaite (PM-Perfect Match) au gène cible et en une sonde ayant une correspondance imparfaite (MM-Miss Match) au gène cible, toutes deux d'une longueur de 25 oligonucléotides. La sonde MM est identique à la sonde PM, à l'exception d'un seul nucléotide au milieu de la séquence (qui est la base complémentaire de la base à cette position sur la sonde PM). Elle permet d'avoir une idée du degré d'hybridation non spécifique pour chaque sonde (figure 2.5).

#### 2.2.3.2 Traitement de données et logiciel MAS5

L'intensité de fluorescence mesurée par un scanner dédié permet une mesure de l'abondance relative de chacun des ARNm présent dans l'échantillon biologique étudié. Cette intensité est mesurée par la moyenne des différences des oligonucléotides PM et MM. Un logiciel spécifique à cette technologie (*MAS d'Affymetrix*) est appliqué pour générer les différents changements des gènes.

Les données générées sont disponibles en format Excel et aussi dans un format compatible avec les applications de statistique *R*.

Les attributs du fichier Excel contiennent quatre valeurs pour chaque groupe de sondes

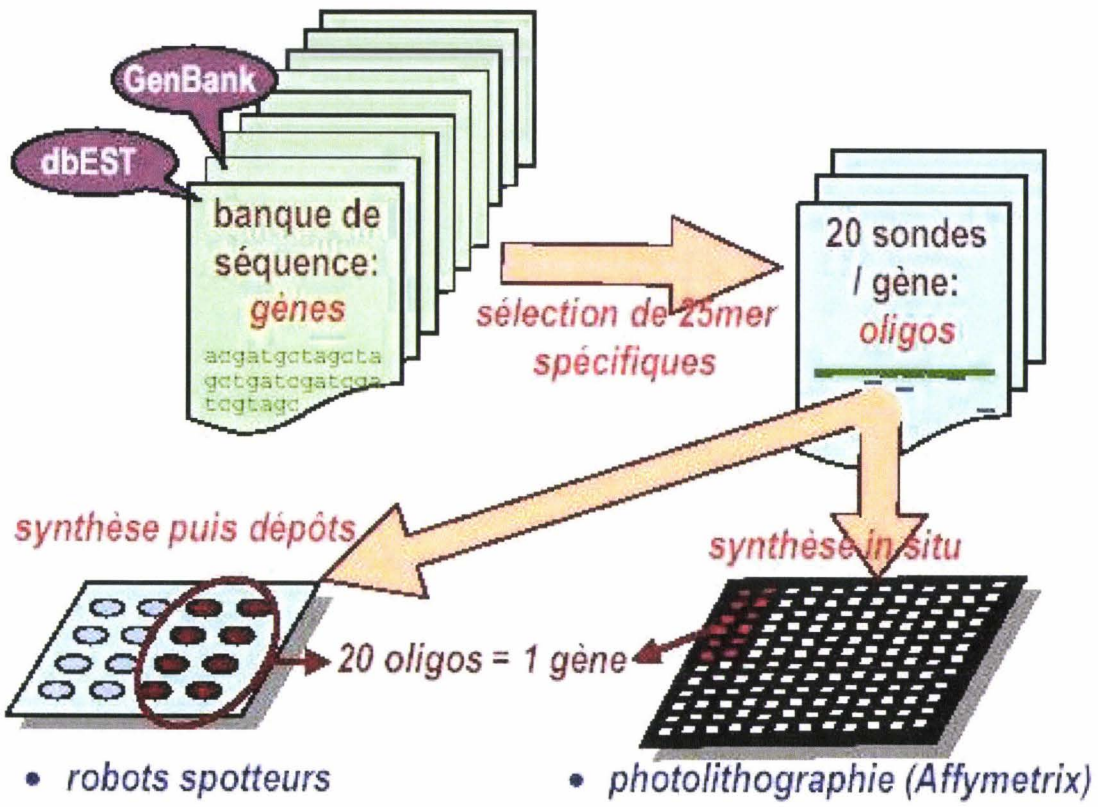



FIG. 2.5 – Technologie Affymetrix

dans une micro-puce. Ces valeurs sont les suivantes :

**probeset** : Identificateur du groupe de sondes.

**Signal** : Valeur d'expression.

**call** : Il existe trois types de classe de détection possibles : P (Présent), A (Absent) ou M (Marginal). La classe indique l'ampleur de l'expression du produit de transcription d'un gène. Si le signal d'expression d'un groupe de sondes ne varie pas significativement de zéro, le produit de transcription reçoit la classe *Absent*. Si une incertitude persiste, la classe *Marginal* est donnée.

**p-value** : La valeur prédictive (p-value) indique pourquoi un gène s'est vu attribuer telle ou telle classe. Elle indique le degré de confiance statistique caractérisant le seuil de détermination de la classe. Il s'agit de l'hypothèse nulle : l'expression moyenne du gène A dans les échantillons traités est la même que celle dans les échantillons témoins. Si l'hypothèse nulle est rejetée, alors il y a une différence significative entre les échantillons traités et témoins dans l'expression du gène A.

Pour définir le seuil au-delà duquel l'écart entre les moyennes sera considéré comme significatif, une valeur $p$ doit être choisie pour le test. La valeur $p$ représente la probabilité qu'une différence observée soit attribuable au hasard. Par exemple, une valeur $p$ de 0,05 signifie qu'il y a une probabilité de 5 % qu'un écart dû au hasard soit considéré comme significatif. En d'autres termes, la probabilité que l'écart observé soit significatif est de 95 %.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | probeset | signal | call | pvalue |
| 2 | 100001_at | 1026.4 | P | 0.000219 |
| 3 | 100002_at | 2.7 | A | 0.697453 |
| 4 | 100003_at | 0.8 | A | 0.956032 |
| 5 | 100004_at | 45.3 | P | 0.001892 |
| 6 | 100005_at | 87.1 | P | 0.000388 |
| 7 | 100006_at | 1.2 | A | 0.765443 |
| 8 | 100007_at | 228.8 | P | 0.001602 |

FIG. 2.6 – Données MAS5.

Un traitement statistique (logiciel *MAS d'Affymetrix*) est appliqué sur les données (entre les échantillons traités et témoins) pour étudier les différents changements des gènes. Les paires (échantillons traités et échantillons témoins) sont comparées afin de détecter et mesurer les changements d'expression génique entre deux groupes de puces. L'analyse compare la différence entre les valeurs (PM-MM) (Perfect Match et MisMatch). Un algorithme de changement (*Change algorithm*) est basée sur la notion de *p-value*. Il permet de générer les différentes statistiques *pvalue*. Selon deux paramètres $\gamma 1$ et $\gamma 2$, un autre

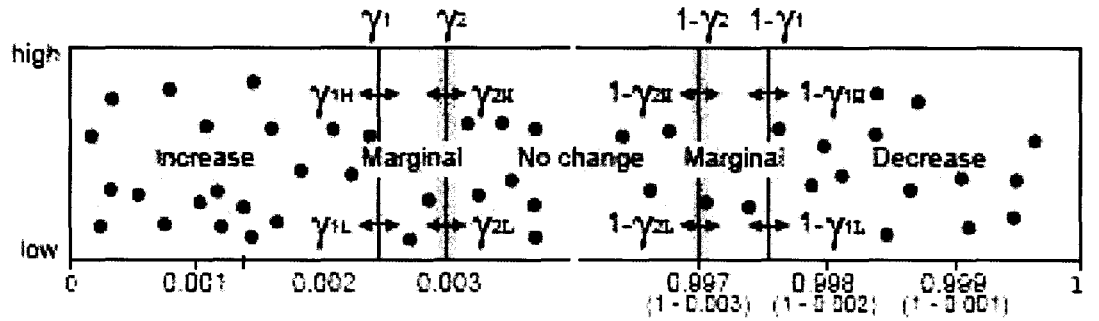



FIG. 2.7 – p-value.

algorithme (Change Call) discrétise les *p-value* en 5 classes : Increase (I), Decrease ($D$), Marginal Decrease ($MD$), Marginal Increase ($MI$) et No Change ($NC$) (figure 2.7).

Pour de plus amples informations sur les classes de détection et les autres algorithmes statistiques utilisés par Affymetrix, le lecteur peut consulter la référence électronique [1].

## 2.3 Banques de données

Les banques de données sont aujourd'hui devenues indispensables pour sauvegarder et structurer les informations issues des expériences de biologie et plus particulièrement des données générées par les différentes technologies de puces à ADN.

Ces banques de données sont plus ou moins généralistes et publiques et directement accessibles par les utilisateurs via le Web. Aussi, il existe de nombreuses banques de données passées dans le domaine privé ou nécessitant l'installation de logiciels en local. Autre intérêt de ces banques de données généralistes est la mise à disposition des jeux de données aux communautés de chercheurs en bio-informatique, mathématiques et statistiques pour le développement de nouvelles méthodologies d'analyse.

Le tableau 2.1 suivant présente quelques références internet pour les bases de données biologiques (biopuce, GO), ainsi que quelques outils d'exploration et d'analyse.

### 2.3.1 Banques de données d'expression de gènes

Parmi les banques de données publiques, les banques de données d'expression de gènes sont particulièrement importantes et intéressantes en terme de partage des connaissances. Ces banques de données se répartissent globalement en 2 catégories plus ou moins généralistes. Les banques de données généralistes pour le dépôt des données d'expression de gènes (repository) ont été développées dans le but de partager les données d'expression de gènes (notamment issues des expériences de puces à ADN) au niveau de la communauté scientifique internationale. L'une de leur priorité est le respect par les biologistes du standard

[1] http ://www.affymetrix.com/support/technical/technotes/statistical_reference_guide.pdf

| Adresse WWW | NOM |
|---|---|
| Microarray : Bases de données | |
| http ://www.ebi.ac.uk/arrayexpress/ | EBI microarray data repository data |
| http ://www.ncbi.nlm.nih.gov/geo/ | Gene Expression Omnibus (GEO) |
| http ://cibex.nig.ac.jp/index.jsp | CIBEX (japon) |
| http ://www.mged.org/ | Microarray Gene Expression Data |
| http ://base.thep.lu.se/ | BASE |
| Microarray : Outils | |
| http ://base.thep.lu.se/plugins/ | BASE plugins |
| http ://www.r-project.org/ | R project |
| http ://bioconductor.org | Projet bioconductor |
| http ://source.stanford.edu/ | SOURCE |
| http ://www.fatigo.org/ | Data mining with Gene Ontology |
| http ://www.affymetrix.com/analysis/ | NetAffx Gene Ontology Mining Tool |
| GO : bases | |
| http ://obo.sourceforge.net | Open Biomedical ontologies |
| http ://www.geneontology.org/ | Gene Ontology |
| http ://www.ebi.ac.uk/GOA | Gene Ontology Annotation (EBI) |
| http ://mged.sourceforge.net/ontologies | MGED Ontology Working Group |
| GO : Outils | |
| http ://www.godatabase.org | AMIGO |
| http ://www.geneontology.org/GO.tools.html | GO Tools, Go annotation |
| http ://vortex.cs.wayne.edu/projects.htm | OntoTools |
| http ://gopubmed.org/ | GO annotation |

TAB. 2.1 – Quelques références internet : Microarray et GO.

international MIAME [BHQ01] pour uniformiser les données et faciliter leur diffusion. Les trois principales banques de données généralistes pour le dépôt des données d'expression de gènes sont ArrayExpress à l'EBI [PSS+04], GEO au NCBI [EDL02] et Cibex (Center for Information Biology gene EXpression database) au DDBJ (DNA Data Bank of Japan) [IIiT+03]. Ces entrepôts sont d'une importance grandissante puisque, aujourd'hui, la majorité des journaux scientifiques requièrent, pour toutes publications dans le domaine des puces à ADN, le dépôt des données d'expression dans au moins une des banques de données publiques conforme au standard international **MIAME**. Les entrepôts permettent de comparer les différentes expérimentations réalisées pour répondre à diverses questions biologiques. Ils offrent la possibilité de confronter des matrices de données d'expression générées par différentes équipes, sur différents modèles et/ou différentes plates-formes. Les résultats de ces comparaisons permettent, entre autre, d'améliorer l'annotation et la connaissance sur les gènes dans les différentes conditions [SSKK03, MMZ+04].

Parmi les plateformes plus développées qui gèrent les expérimentations par puce à ADN, on trouve BASE (BioArray Software Environment) [STVC+02].

En effet, BASE [2] est une base de données permettant de gérer l'importante quantité de données générées par des analyses de puces à ADN. BASE gère les informations biologiques, les données brutes et les images. BASE possède également des outils de normalisation, de visualisation et d'analyse des données.

### 2.3.2 Gene Ontology (GO)

La technologie des puces à ADN offre un aperçu des corrélations entre les gènes et les phénomènes biologiques mais elle ne permet pas à elle seule de révéler la causalité des mécanismes de régulation. Aussi, l'intégration des informations issues de différentes sources contrôlées comme les ontologies, les résumés d'articles scientifiques ou les banques de données protéiques, est devenue indispensable pour interpréter les résultats d'analyse des données issues des expériences génomiques.

Gene Ontology [3] (appelé GO) a été développé par le Gene Ontology Consortium, groupe de travail international basé à l'EBI, pour aider à l'annotation des génomes [ABB00b]. Son objectif est d'établir un vocabulaire structuré, contrôlé et dynamique pour décrire le rôle des gènes et produits de gènes. Gene Ontology est devenu un standard pour l'annotation des génomes. Cette ontologie facilite le partage des connaissances. Par ailleurs, elle permet de mettre à jour les manques dans la connaissance actuelle et d'interpréter les résultats d'analyse des données issues des expérimentations de puces à ADN.

GO se compose de trois ontologies qui définissent les processus biologiques, les fonctions moléculaires et la localisation cellulaire des produits de gènes. Le processus biologique fait référence à l'objectif biologique auquel un gène ou produit de gène participe (e.g. la croissance cellulaire ou la transduction du signal). Un processus biologique est le résultat d'une ou plusieurs fonctions moléculaires associées dans un ordre donné. La fonction moléculaire décrit l'activité biochimique ou l'action du produit d'un gène (e.g. enzyme transporteur).

[2] http ://base.thep.lu.se
[3] http ://www.geneontology.org

La localisation cellulaire présente l'endroit de la cellule où se trouve la forme active du produit d'un gène.

Les trois ontologies GO sont structurées sous forme d'un graphe orienté acyclique ou DAG (Directed Acyclic Graph). Ce DAG (figure 2.8) est un réseau où chaque noeud représente un terme GO. Chaque terme GO peut être un *enfant* de un ou plusieurs *parents*. Un gène peut avoir plusieurs produits et les produits d'un gène possèdent une ou plusieurs fonctions biochimiques. Le terme *enfant* est toujours plus spécifique que le ou les termes parents. La relation entre un *enfant* et son *parent* peut être du type *est un(e)(is_a)*, identifié par un pourcentage, lorsque le terme enfant est une instance du terme parent. Elle peut aussi être de la forme *fait parti de (part_of)*, représentée par $<$ , si le terme enfant est un élément du parent. Si un terme a plusieurs *parents*, il peut avoir différentes relations avec chacun de ses *parents*.

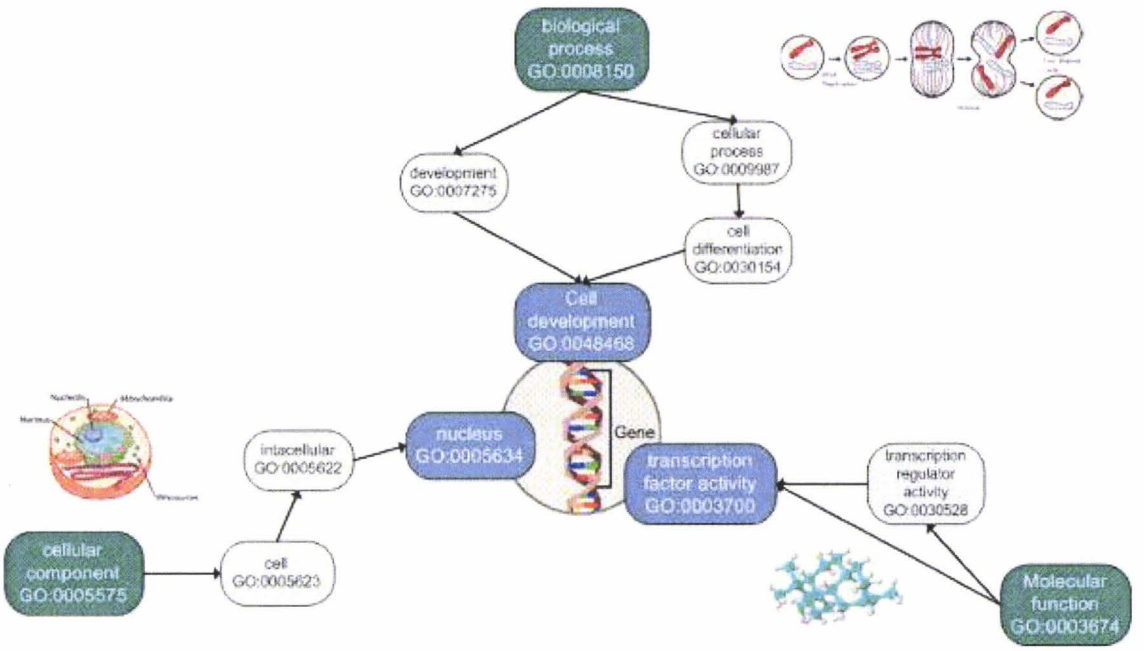



FIG. 2.8 – Graphe orienté acyclique de Gene Ontology.

## 2.4 Applications

Les domaines d'application des puces à ADN sont très larges et intéressent de nombreux secteurs tels que la recherche biologique (et notamment la génomique fonctionnelle), pharmaceutique, le génotypage, le diagnostic, etc.

### 2.4.1 Analyses d'expression de gènes

Les premières puces ont servi à évaluer l'expression simultanée de milliers de gènes dans des systèmes biologiques bien connus, tels que celui du métabolisme respiratoire et de fermentation chez la levure [DVI97], le cycle cellulaire de la levure [CCW98] et la stimulation de fibroblastes par le sérum [IER99]. Ces premiers travaux ont permis de valider la technologie.

### 2.4.2 Action des médicaments

La pharmacogénomique consiste à identifier les gènes impliqués dans l'efficacité (ou l'inefficacité) d'un produit, ou ses effets indésirables. Elle conduit à une meilleure compréhension des mécanismes d'action des médicaments. En montrant qu'une molécule a sur une cible une action variable, la biopuce ouvre le champ des potentiels thérapeutiques. Elle permet aussi d'identifier les effets secondaires d'un produit et, lors des essais cliniques, de faire des mesures de toxicité.

Les puces à ADN peuvent ainsi être utilisées pour étudier le mécanisme d'action d'un médicament. La plupart des médicaments agissent en inhibant leur molécule cible, enzyme ou récepteur. Par conséquent, une mutation du gène correspondant devrait avoir un effet similaire sur le transcriptome de la cellule.

Marton et al. [MDB98] ont utilisé une puce à ADN contenant l'ensemble des gènes de la levure afin de démontrer l'existence d'une corrélation significative entre le profil obtenu lors d'une stimulation médicamenteuse antimicrobienne et le profil d'expression d'une levure portant un gène muté et impliqué dans le métabolisme d'action de ce médicament.

### 2.4.3 Analyses d'ADN génomique

L'étude des variations génomiques est d'une grande utilité en recherche bio-médicale. Une grande partie de la variabilité inter individuelle observée au sein d'une même espèce, en particulier la susceptibilité à certaines maladies, est dûe à des différences (ou polymorphismes) existant au niveau de la séquence de l'ADN génomique. En raison de leur potentiel et de la rapidité d'analyse qu'elles présentent, les puces à ADN offrent un avantage considérable sur les techniques déjà existantes pour aborder ces variations.

#### Génomiques comparées :

Behr et al. [BWW99] ont étudié plusieurs souches du bacille de Calmette et Guérin (BCG), originaires de l'Institut Pasteur au cours du XXe siècle. A partir de la séquence complète de *M. tuberculosis*, un assemblage de sondes génomiques couvrant l'ensemble de ce génome a été déposé sur une puce à ADN. Le génome de chaque souche de BCG étudiée a été amplifié, marqué par un fluorochrome et hybridé sur la puce à ADN. Cela a permis d'identifier des divergences au niveau génomique et de les corréler de façon temporelle avec l'historique de la dissémination mondiale de ces souches. Cette étude suggère l'existence

d'un lien entre l'efficacité du vaccin contre le BCG et la prééminence des souches classifiées par ces marqueurs génomiques.

**Reséquençage et détection de SNP :**

Grâce aux puces à ADN, une séquence connue peut être reséquencée. Il faut donc créer une puce à ADN contenant tous les 25 mers chevauchants, définissant la séquence à interroger ainsi que trois amorces contenant les trois permutations possibles pour le nucléotide central de l'amorce (c'est-à-dire qu'un T sera remplacé par un A, un C et un G). Cette méthode a été utilisée par Wang et al. [WFS98] pour reséquencer 2,3 Mb du génome chez 7 individus et a ainsi permis d'identifier 3 241 SNP humains. Cette approche a également été utilisée afin d'identifier des polymorphismes dans un gène de susceptibilité au cancer du sein (BRCA1) [HBC+96] ainsi que pour plus d'une centaine d'autres gènes impliqués dans des processus vasculaires, métaboliques et endocriniens.

**Génomique fonctionnelle :**

La plupart des applications en génomique fonctionnelle consistent à classer les patients ou les situations biologiques suivant la fonction de gènes. Il est possible de faire l'inverse pour explorer la fonction des gènes et déterminer la fonction d'un gène inconnu. Ainsi, à partir d'un compendium de profils d'expression chez C. elegans, quelques grandes fonctions de gènes ont été identifiées [KJK+01].

Des développements similaires commencent à apparaître pour l'analyse massivement parallèle des protéines ou d'autres composantes cellulaires.

## 2.5 Différentes techniques de datamining pour les puces à ADN

Les résultats de puces à ADN sont des matrices représentant les niveaux d'expression des gènes sous plusieurs conditions étudiées. A chaque gène est associé un profil d'expression. Le but de l'analyse est d'identifier les gènes ayant des profils semblables (ou responsable des phénomènes étudiés), ces gènes peuvent être régulés par les même facteurs ou intervenir dans le même processus biologique.

En raison du nombre important de gènes et de la complexité des réseaux géniques, les techniques de dataming, d'apprentissage et de statistique sont avérées un outil très utile pour l'analyse de profils d'expression. Parmi les techniques utilisées, nous citons la classification (et clustering), les réseaux de neurones, l'analyse en composante principale (ACP) et règles d'association.

L'un des objectifs de l'analyse des données d'expression consiste à classer les profils en fonction de leur différence d'expression selon certains facteurs biologiques en $K$ classes (tumeur, cancer, type de bactérie ...).

Les algorithmes de classification sont définis comme des méthodes de répartition d'un ensemble d'objets (points ou vecteurs) en plusieurs sous-ensembles, sur la base de leurs similarités ou dissimilarités. Le but est de construire des groupes qui minimisent la variabilité intra-groupe tout en maximisant les distances inter-groupes. Plus précisément, ils visent à trouver l'ensemble des groupes (gènes ou échantillons) dont les membres sont très similaires mais distants des autres membres sur la base de leur profil d'expression. Les algorithmes de classification se regroupent en deux grandes catégories : les approches supervisées et non supervisées. Les méthodes non supervisées groupent les objets sans a priori. Ces techniques sont dites exploratoires et sont essentiellement employées pour la découverte de classes. A l'inverse, les méthodes supervisées utilisent de la connaissance a priori.

Parmi les méthodes utilisées, citons la classification hiérarchique, les cartes topologiques de Kohonen et les méthodes dites des nuées dynamiques. Dans ce cadre, les problèmes de choix du nombre de classes et de méthodes de classification deviennent des problèmes de choix de modèles, et il n'existe pas de méthode générique pour choisir le nombre de classes ni la meilleure méthode de classification (cluster, gene cluster, ...).

### 2.5.1 Classification supervisée

Il existe de nombreuses méthodes de classification supervisée (analyse discriminante, analyse de voisinage, machines à support vectoriel (SVM), etc.) permettant de prédire la classe du gène, l'objectif étant de minimiser le taux d'erreur de prédiction.

#### L'analyse discriminante

Cette méthode, issue des statistiques, part de la connaissance de K classes des gènes et cherche les combinaisons (linéaires, quadratiques) des expressions des gènes (variables) en améliorant la discrimination entre les classes [DFS02].

#### Les Support Vector Machines (SVM)

Ce sont des méthodes de classification supervisée. Le principe général est une séparation d'un ensemble de données de test (d'apprentissage) par un hyperplan maximisant la distance aux points de test. Dans le cas où aucune séparation n'est possible, il y a la possibilité de coopération entre $SVM$. Les nouvelles données sont testées par rapport à l'hyperplan séparateur et classées avec un intervalle de confiance. Cette méthode a été utilisée pour la classification de gènes [PWCG01] et de tissus [FCD+00] et pour l'étude de pathologie [GWBV02].

#### Les méthodes des plus proches voisins ou KNN (K Nearest Neighbor)

Cette méthode permet de prédire la classe d'un nouveau gène en fonction de la classe la plus représentée parmi ses $K$ plus proches voisins. Cette méthode dépend du choix de

nombre de voisins $K$, de la métrique utilisée et de l'ensemble de test (apprentissage).

#### Analyse en composante principale (ACP)

L'analyse en composante principale ou ACP (PCA - Principal Component Analysis) est une méthode statistique pour l'exploration de données multi-variées. Son objectif est de réduire la dimension de l'espace des données en déformant le moins possible la réalité. Pour cela, elle détermine une suite d'axes orthogonaux, non corrélés, conservant au mieux les distances entre les individus. Généralement, les composantes principales utilisées sont les 2 ou 3 premières puisqu'elles témoignent des principales variations observées dans le jeu de données original. Les dernières composantes reflètent quant à elles les bruits résiduels. Dans notre cas, chaque gène est représenté par un point dans un espace de dimension $N$ ($N$ est le nombre d'expérimentations). Cette méthode permet de réduire l'espace de l'analyse. Par exemple, la réduction de la matrice des données gènes/expériences en vecteurs propres permet la mise en évidence de différents regroupements [ABB00a, HMM+00].

### 2.5.2 Classification non supervisée

Les méthodes de classification non supervisée sont des techniques de regroupement (clustering) où un processus automatique sépare les données observées en groupes distincts sans aucune connaissance préalable des classes existantes.

Les algorithmes de clusterisation groupent les gènes en fonction de leur profil d'expression basée sur une métrique qui calcule la similarité entre deux profils [ESBB98]. La plupart des algorithmes utilisent le coefficient de corrélation statistique ou la distance Euclidienne. Ce sont les algorithmes les plus souvent utilisés pour l'analyse des données pour les puces à ADN.

#### La méthode des Self-Organizing Maps (SOM)

Elle procède en faisant une série de partitions, chacune d'elles ayant un vecteur de référence contenant autant de points qu'il y a d'expériences considérées. Pour affecter un gène à une partition, un gène est tiré au hasard et il est associé au vecteur de référence dont il est le plus proche. Ce vecteur est ensuite ajusté pour augmenter sa similarité par rapport au vecteur d'expression du gène. Ensuite, les vecteurs des partitions sont à leur tour ajustés. Ces étapes sont répétées plusieurs de fois. A la fin, les gènes sont placés dans la partition choisie en fonction de la similarité des vecteurs de références.

#### La méthode des nuées dynamiques (ou K-means clustering)

La méthode K-means est assez similaire à la méthode des SOM, car elle utilise la partition, mais ici, une partition n'influence pas directement les autres. On peut considérer que cette technique est uni-dimensionnelle. Tout d'abord, les vecteurs sont initialisés au hasard, puis les gènes sont séparés avec les vecteurs qui leur sont les plus similaires. Ensuite,

chaque vecteur de référence est recalculé en faisant une moyenne des gènes qu'il contient. Ces étapes sont répétées jusqu'à convergence.

### Classification Hiérarchique

Les méthodes de classification hiérarchique sont aujourd'hui les techniques de classification non supervisée les plus utilisées pour étudier les profils d'expression de gènes ou d'échantillons. Elles génèrent des suites de classes emboîtées qui définissent une hiérarchie de partitions encore appelée classification hiérarchique.

Ce type de classification repose sur la construction hiérarchique d'un arbre. Le modèle s'applique jusqu'à ce que tous les profils individuels et que tous les noeuds soient joints pour former un arbre hiérarchique unique. L'avantage de cette approche est sa simplicité, et le résultat final est facilement visualisable. Cette méthode fournie une hiérarchie de partition des gènes. En regardant l'arbre, on peux retrouver l'ordre dans lequel les gènes ont été regroupés.

L'application des algorithmes de classification supervisée ou non supervisée aux données de puces à ADN pose un problème : chaque gène étant considéré comme une variable, le nombre de variables (environ 10000) est trop grand comparé au nombre d'observations disponibles (environ 100). Le classificateur construit risque donc d'être surajusté, ce qui rend sa performance médiocre. Pour réduire le nombre de variables, on peut alors utiliser des méthodes de compression (type ACP ou ondelettes), qui consistent à créer de nouvelles variables synthétisant l'information contenue dans les variables initiales. Le défaut de ces approches est qu'elles sont généralement non dédiées : dans les méthodes usuelles de compression, la construction des nouvelles variables se fait sans prendre en compte l'objectif de classification.

## 2.6 Conclusion

Les techniques de classification pour l'analyse de puces à ADN ont été largement utilisées pour identifier des groupes de gènes partageant des profils d'expression similaires et les résultats obtenus sont très concluants. Néanmoins, ces méthodes ne permettent de découvrir qu'une partie des relations parmi toutes les relations potentielles entre les gènes, les classes recherchées doivent être vérifiées sur l'ensemble des expérimentations et un gène ne peut appartenir qu'à une seule classe. En effet, ces méthodes ne donnent pas la relation exacte qui peut exister entre deux gènes ou deux groupes, et ne permettent de donner qu'une image globale, l'information à un niveau plus local pouvant alors être perdue.

C'est pourquoi nous proposons d'utiliser les règles d'association. En effet, cette méthode permet de mettre en évidence des relations plus précises entre les gènes. La recherche de règles d'association est une méthode classique de datamining mais avait peu été appliquée sur ce type de données. En effet la plupart des approches utilisées pour l'étude d'expression génique ont consisté jusqu'à présent à de la classification supervisée ou non supervisée.

Nous allons proposer dans le chapitre suivant une approche de résolution multi-objectif

pour les règles d'association.

# Chapitre 3

# Modélisation multi-objectif des règles d'association

Sommaire


3.1 Introduction . . . . . 31
3.2 Datamining . . . . . 32
3.3 Règle d'association . . . . . 34
3.4 Mesures de qualité existantes . . . . . 40
3.5 Propriétés des bonnes mesures . . . . . 46
3.6 Notre approche : Analyse statistique des critères . . . . . 50
3.7 Conclusion . . . . . 59


Ce travail a fait l'objet d'une publication dans la revue RAIRO Operations Research, Special Issue on Cooperative methods for Multiobjective Optimization [KDT06a], et de présentations aux conférences Cinquième Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la décision ROADEF'2003 [KDT03a] et SIAM Bioinformatics Workshop, in conjunction with fourth SIAM International Conference on DataMining [KDT04a] et Workshop on Real-life applications of Metaheuristics [KDT03b].

## 3.1 Introduction

La problématique des règles d'association est si classique et utilisée que différentes communautés scientifiques (statistique, apprentissage, datamining, optimisation, ...) ont proposé leurs mesures d'évaluation. Le nombre de règles obtenues par les algorithmes classiques utilisés en extraction de connaissances est très important, ce qui ne permet pas ensuite aux utilisateurs de sélectionner les règles les plus pertinentes. Une question fondamentale est donc : qu'est-ce qu'une règle pertinente ? Comment évaluer la qualité d'une règle ?

La qualité d'une règle peut dépendre de plusieurs caractéristiques (sa force de prédiction, son nombre d'occurrences, ...) en fonction du contexte. Cela a donné lieu à la définition d'un grand nombre de critères mesurant cette qualité. Dans une approche monocritère, on cherche l'ensemble des règles qui optimisent une unique mesure qui satisfait certaines propriétés, par contre dans une approche multicritère, on cherche un ensembles des règles qui optimisent un ensemble restreint de mesures complémentaires et indépendantes.

Dans ce chapitre, nous introduirons dans un premier temps le data mining et ses différentes techniques. Nous présenterons ensuite la recherche de règles d'association et les méthodes de résolution de la littérature. Puis, nous aborderons les différents indicateurs mesurant la qualité des règles d'association (les plus utilisés). Nous évoquerons les propriétés et préférences proposées par différents auteurs tels que Tan et al. [TKS02], Hilderman [HH99] et Piatetsky-Shapiro [PS91] et nous terminerons par létude statistique de ces critères que nous avons réalisée, en vue de déterminer le ou les critères à utiliser lors d'une approche par optimisation combinatoire multi-objectif [KDNT03, KDT04a].

## 3.2 Datamining

Durant ces dernières années, on assiste à une forte augmentation tant dans le nombre que dans le volume des informations mémorisées par des bases de données scientifiques, économiques, financières, administratives, médicales, etc. Le stockage en lui-même ne pose pas de réelles difficultés du point de vue informatique, mais le besoin d'interpréter ou de trouver de nouvelles relations entre les éléments stockés dans ces bases a suscité beaucoup d'intérêt. Ainsi, la mise au point de nouvelles techniques informatiques est devenue un thème important pour bon nombre de chercheurs. Le "Knowledge Discovery" et le "Data Mining" représentent un domaine émergeant essayant de répondre à ces objectifs.

Knowledge Discovery in Databases (KDD) ou Extraction de Connaissance à partir de Données (ECD) : processus non trivial d'identification de motifs (patterns) valides, nouveaux, potentiellement utiles et compréhensibles à partir d'une grande collection de données [FPSS96].

Le processus de ECD est itératif et interactif et implique plusieurs étapes (voir figure 3.1)

- Définir le domaine d'application,
- Créer l'ensemble des données ciblées par cette application,
- Pré-traiter et nettoyer les données,
- Réduire et transformer les données afin de trouver les attributs utiles selon les questions posées,
- Choisir l'algorithme de fouille,
- Interpréter les motifs découverts.

Le data mining, en français la fouille de données (exploration de données) est une étape dans le processus du KDD. Le datamining est un processus itératif de découverte de modèles d'une base de données. Il s'agit donc d'extraire des connaissances à partir de ces données afin de décrire le comportement actuel et/ou de prédire le comportement futur d'un procédé. Le terme de Data mining est souvent utilisé comme un synonyme de KDD,

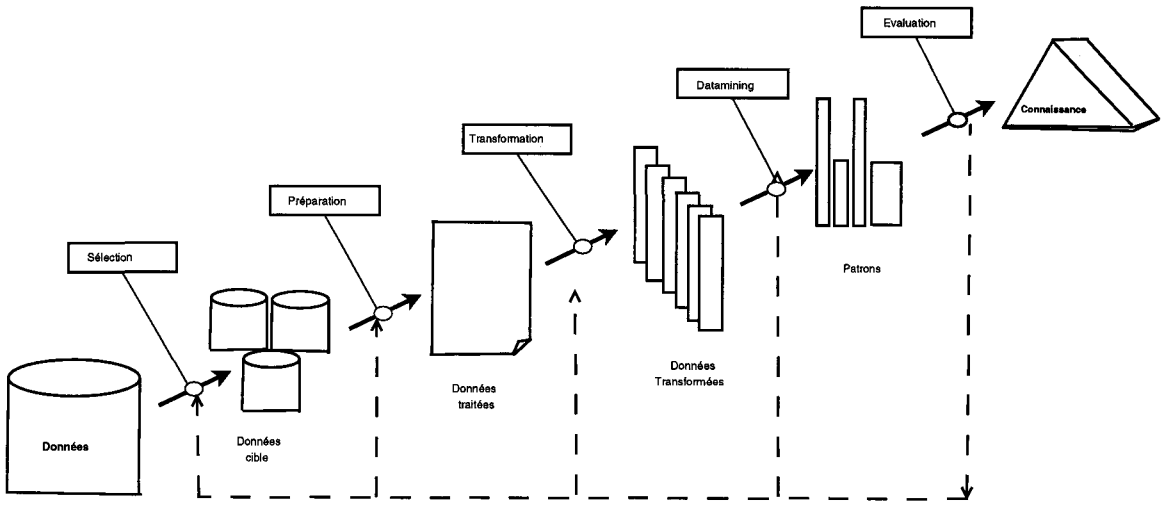



FIG. 3.1 – Processus de l'extraction de connaissance.

bien que la plupart des chercheurs voient en lui une étape essentielle de la découverte de connaissances.

Parmi les différentes tâches de datamining les plus utilisées, on trouve :

**La classification** : système d'apprentissage supervisé qui développe un modèle interne de concepts à partir d'exemples pour classer de nouveaux objets.

**Le clustering** : il permet d'identifier des distinctions conceptuelles dans un grand volume de données. C'est un système de classification non supervisé, qui utilise le principe de maximisation de la similarité intra-groupe et de minimisation inter-groupes pour classer les données en clusters.

**La sélection d'attributs** : En général la base de données est très dense, composée de plusieurs milliers d'attributs. Le temps de traitement est très grand. La sélection d'attributs consiste à déterminer le sous ensemble optimal suffisant d'attributs.

**La recherche des règles d'association** : consiste à extraire un ensemble de formules logiques conditionnelles qui déduisent la valeur d'un attribut but à partir des valeurs d'autres attributs apparaissant simultanément

L'application des techniques de datamining à la biologie constitue aujourd'hui un champ de recherche en émergence. En biologie moléculaire, l'analyse du transcriptome a conduit au stockage d'un grand nombre de données sur l'expression des gènes, notamment grâce à la technique des puces à ADN, présentée au chapitre 2.

Nos travaux se situent principalement sur la recherche de règles d'association pour les puces à ADN.

## 3.3 Règle d'association

### 3.3.1 Problématique

Le problème de la recherche de règles d'associations a été introduit par Agrawal, Imielinski et Swami [AS94]. En entrée, on dispose d'un ensemble de transactions où chacune d'entre elles est un ensemble de littérales appelées Items. Une règle d'associations peut être la suivante : "<30% des transactions qui contiennent de la bière et des noisettes contiennent aussi de la moutarde ; 2% de toutes les transactions contiennent ces trois éléments ". Ainsi, la valeur 30% représente la confiance de la règle et 2% le support.

La formulation du problème est la suivante : Une règle d'associations est une implication sous la forme A ==> B. La partie gauche est l'antécédent, et la droite, le conséquent. Plusieurs antécédents et conséquents peuvent constituer une règle. La façon intuitive d'interpréter une règle est que les transactions qui contiennent A tendent à contenir B, c'est-à-dire en reprenant l'exemple, la plupart des fois où de la bière et des noisettes sont achetées, de la moutarde l'est également. Cette règle représente le comportement des clients.

La recherche des règles d'association consiste à extraire un ensemble de formules logiques conditionnelles qui déduisent la valeur d'un attribut but à partir des valeurs d'autres attributs apparaissant simultanément. Un exemple concret serait de déterminer les articles qui se trouvent simultanément sur un ticket de caisse. Ainsi, nous pourrions obtenir cette règle extraite d'une base de tickets de caisse :

- 70 % des tickets qui contiennent A et B contiennent également C.
- 90 % des clients qui achètent du beurre et du pain achètent aussi du lait.

Soit $I = \{I1, I2, I3, ..., In\}$ un ensemble d'items ou attribut. Une transaction T est définie comme un sous-ensemble d'items dans I, chaque transaction dans une base de donnée BD est nommée par un identifiant (TID).

Une règle d'association R est une implication de la forme C⟶P (IF C THEN P avec C $\subset$ I, P$\subset$I et $C \cap P = \phi$. La force d'une règle est mesurée par deux indicateurs principaux : le support et la confiance. Une règle est plus ou moins vraie pour un certain pourcentage constituant le support de la règle.

Le support est définie par la probabilité P(C, P).

Si support(R) est supérieur à un minimum (minsup) alors la règle R est dite fréquente.

Le support n'est pas suffisant pour évaluer la force d'une règle. Il se peut en effet qu'une règle corresponde à des apparaissent rarement, mais soit très valide. Par exemple, dans l'étude des habitudes achats, lorsqu'on achète une brosse à dents, on achète très souvent du dentifrice. Cependant on change rarement de brosse à dents. Donc, la règle brosse-à-dents ⟶ dentifrice a un faible support, mais elle est très valide ! Pour cela la notion de confiance a été introduite. La confiance correspond à la probabilité conditionnelle P(P/C) qui est la probabilité a posteriori que P soit vraie étant donné que C est vraie.

Le problème de la définition des règles d'associations (avec l'approche de Apriori) consiste à générer toutes les règles d'association qui ont un support et une confiance supérieux aux minima spécifiés par l'utilisateur (minsup, minconf respectivement).

La recherche de règles d'associations se décompose en deux principales étapes :

- Trouver tous les ensembles d'objets (itemsets) qui ont un support supérieur au support minimum. Les itemsets qui atteignent le support minimum sont appelés les ensembles d'objets fréquents (itemsets fréquents). L'espace de recherche d'énumération de tous les itemsets est $2^m$, où $m$ est le nombre d'items. On peut représenter tous les itemsets sous forme d'un treillis. La figure 3.2 montre le treillis des itemsets pour 5 items ABCDE.
- Utiliser les ensembles d'itemsets fréquents pour déduire les règles recherchées vérifiant le critère de confiance. Ainsi, si ABCD et AB sont des ensembles d'objets fréquents, alors on peut savoir si la règle AB $\longrightarrow$ CD convient en calculant le ratio conf $= \frac{support(ABCD)}{support(AB)}$. Si conf $\geq$ minconf, alors la règle est retenue (la règle aura forcément un support minimum car ABCD est fréquent). La complexité de génération des règles est $O(r - 2^l)$, où $r$ est le nombre d'itemsets fréquents et $l$ la longueur de l'itemset le plus long.

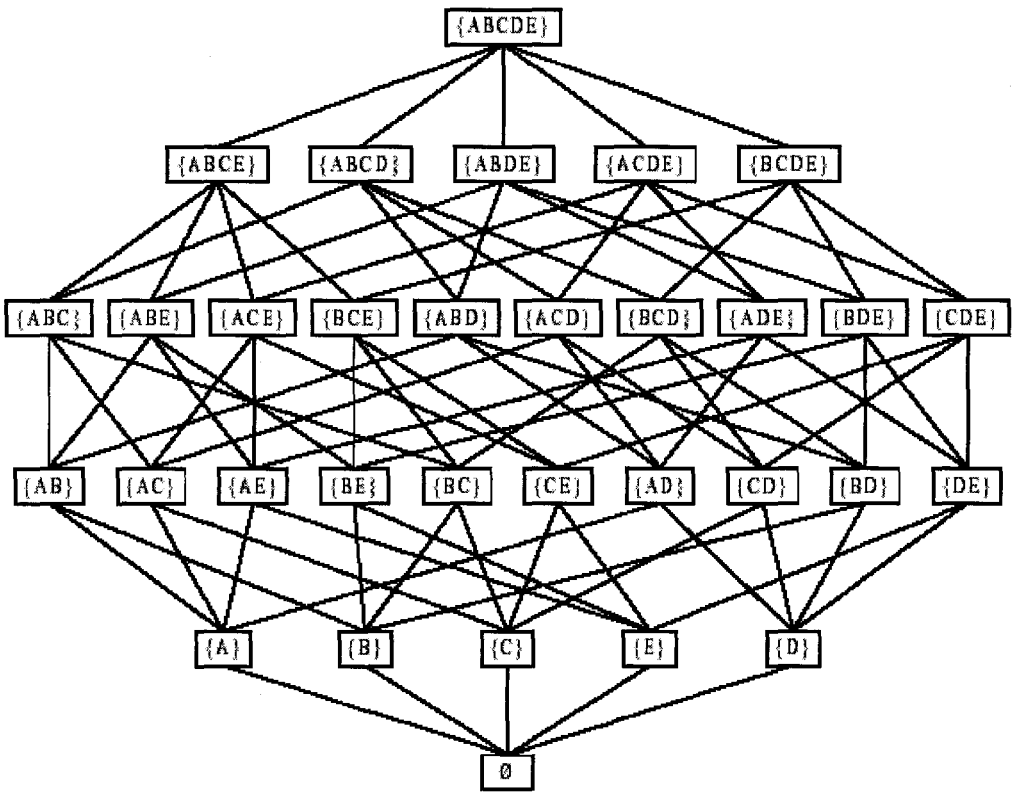



FIG. 3.2 – Exemple de treillis pour 5 items A B C D E

Comment optimiser les calculs des indicateurs support et confiance sur de larges bases de données ? Le principe est d'élaborer des stratégies pour parcourir l'espace de recherche en essayant de réduire l'exponentialité et l'accès aux données. Différents algorithmes sont

présentés dans la suite, pour la recherche de l'ensemble d'items fréquents.

### 3.3.2 Algorithmes pour les règles d'association

Les algorithmes traditionnels effectuent de multiples boucles à partir des données pour découvrir les ensembles d'items fréquents, puis ils utilisent un filtrage sur la confiance afin de retenir uniquement les règles satisfaisantes.

La figure 3.3 présente une classification non exhaustive des différents algorithmes.

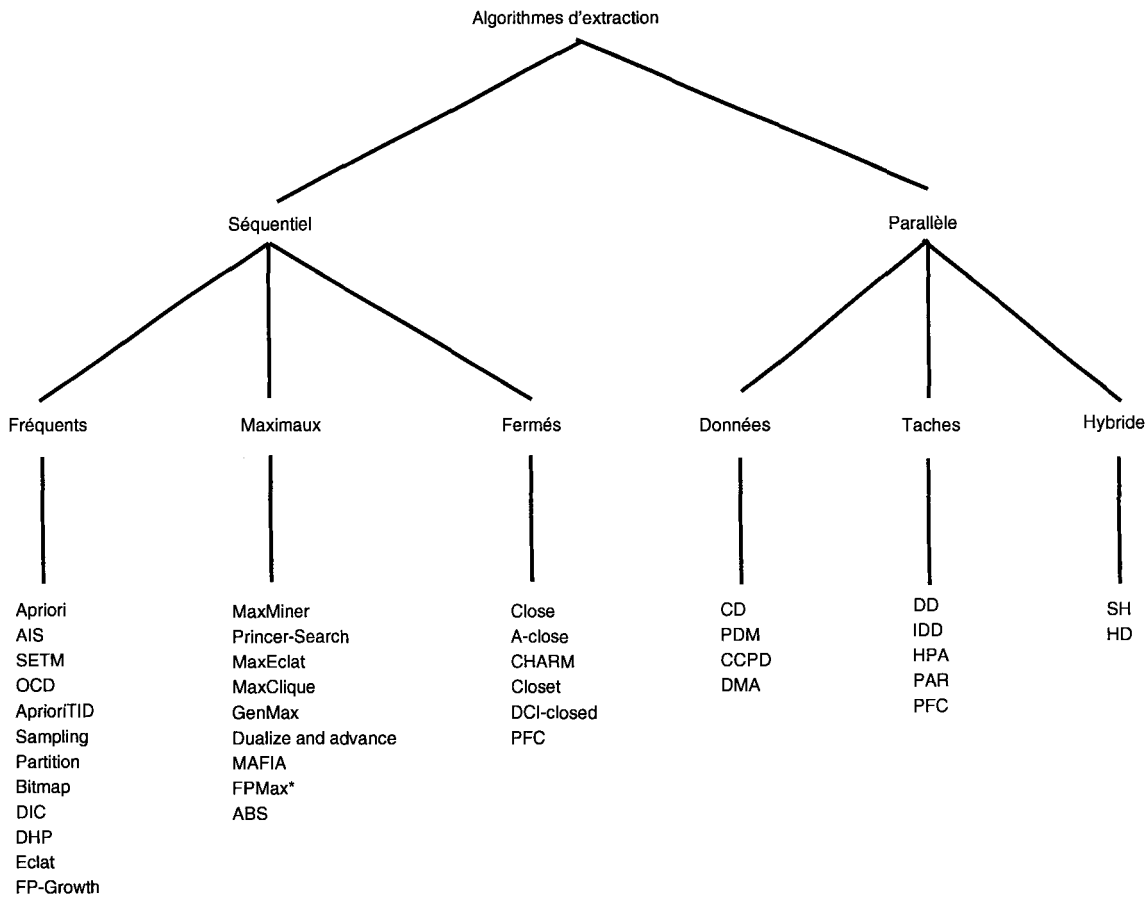



FIG. 3.3 – Classification des algorithmes de recherche de règles d'association basés sur les itemsets.

Les algorithmes AIS [AIS93] et SETM [HS95] sont les travaux précurseurs pour l'extractions des itemsets fréquents. L'algorithme OCD a été proposé de manière indépendante de Apriori, à quelques mois d'interval et fonctionnent de la même façon [MTV94]. Plusieurs optimisations et structures de données permettant d'améliorer l'efficacité de ces deux algorithmes ont été proposées. Parmi celles-ci, on peut citer AprioriHybrid [AS94],

DHP [PCY95], Sampling [Toi96], Partition [SON95], DIC [BMUT97].

Certains de ces algorithmes sont brièvement décrits ci-après :

#### 3.3.2.1 Algorithme Apriori

L'algorithme Apriori [AS94] est le premier algorithme proposé. Il est devenu l'algorithme de référence dans le domaine, ayant donné lieu à de nombreuses améliorations.

Il procède en plusieurs passes (voir l'algorithme 3.3.2.1). Dans la première passe les 1-itemsets (ensembles d'items de taille 1) fréquents sont cherchés. Lors des passes suivantes, l'algorithme utilise une procédure Apriori-gen pour générer un ensemble fréquent candidat de taille k à partir de deux ensembles fréquents de taille k-1. L'algorithme s'arrête quand il n'est plus possible de générer de nouveaux candidats de taille supérieure.

**Algorithme 1** Algorithme de recherche des itemsets fréquents

$L_1 = \{$1-itemsets fréquents$\}$
$k \leftarrow 2$
**tantque** $L_{k-1} \neq \phi$ **faire**
  $C_k =$ apriori_gen($L_{k-1}$)
  **pour tout** *instance* $t \in T$ **faire**
    $C_t =$ subset($C_k$,t)
    **pour tout** candidat $c \in C_t$ **faire**
      c.count++ ;
    **fin pour**
  **fin pour**
  $L_k = \{ c \in C_k$ / c.count $\geqslant$ MINSUP $\}$
  k++
**fin tantque**
L $= \cup_i L_i$
return L

#### Génération des itemsets candidats : Algorithme APRIORI-GEN

Cet algorithme construit à partir de l'ensemble des (k-1)-itemsets fréquents Lk-1, l'ensemble des k-itemsets candidats Ck. Il se déroule en deux étapes (voir l'algorithme 2) :

- Il fusionne deux (k-1)-itemsets P et Q qui partagent leur (k-2)-premiers items.
- Il supprime de Ck tout itemset X pour lequel au moins un sous-ensemble de longueur k-1 de X n'appartient pas à Lk-1.

#### 3.3.2.2 Autres algorithmes

L'efficacité de Apriori diminue en présence de données denses ou fortement corrélées. Il existe de nombreux algorithmes de recherche de règles d'association proposant des améliorations pour accélérer la construction des ensembles fréquents (voir la figure 3.3 qui présente une classification non exhaustive des différents algorithmes).

**Algorithme 2** APRIORI-GEN

```
Ck = { }
pour tout chaque X de Lk-1 faire
  pour tout chaque Y de Lk-1, avec X < Y, X et Y partagent leur k-2 premiers items
  faire
    pour tout  chaque Z ⊂ X ∪ Z tel que : |Z |= k-1  faire
      si Z ⊈ Lk-1 alors
        continuer avec Y suivant ;
      finsi
    fin pour
    Ajouter (X ∪ Z) à Ck ;
  fin pour
fin pour
Return Ck
```

Certains de ces algorithmes sont brièvement décrits ci-après :

**AprioriTID**

AprioriTID est une variante de l'algorithme Apriori proposé par Agrawal et al. [AS94]. Il permet de diminuer progressivement la taille de la base de données considérée, dans le but de la stocker en mémoire et de ne plus réaliser d'opération d'entrée / sortie, après le premier parcours de celle-ci. En effet, on met toute la base en mémoire et à chaque niveau du treillis, on représente les transactions par les k-itemsets candidats qu'elle contient ; une seule passe suffit donc, mais il faut que toute la base tienne en mémoire.

**DIC (dynamic itemset counting)**

DIC (Dynamic Itemset Counting) a été proposé par Brin et al. en 1997 [BMUT97]. Cet algorithme procède par niveaux, mais au niveau k dès qu'un itemset a atteint le seuil de fréquence, on introduit les itemsets candidats de niveau k + 1 qu'il contribue à générer, ce qui diminue le nombre de passes nécessaires sur la base.

**A-Priori Partition :**

Savasere et al. [SON95] propose une extension d'Apriori où la base de données est divisée en N partitions qui tiennent chacune en mémoire. Les partitions peuvent ainsi être facilement traitées en parallèle. Chaque partition est traitée indépendamment et on réalise la découverte des ensembles fréquents pour chaque partition. On remarque qu'un ensemble fréquent doit l'être dans au moins une partition.

**Sampling :**

Il se base sur l'extraction d'un échantillon de la base qui tienne en mémoire, à partir duquel on construit l'ensemble des itemsets fréquents dans l'échantillon ainsi que sa bordure négative constituée des itemsets non fréquents minimaux dont toutes les parties sont fréquentes, ce qui limite le risque de non exhaustivité [Toi96].

**Algorithme bitmap :**

L'algorithme Bitmap (Bitmap algorithm) [Gar99] se base sur la notion d'index bitmap. Les calculs pour les opérations d'unions et d'intersections sont plus rapides dans les vecteurs binaires que dans les listes. L'index bitmap est une matrice binaire de N lignes et P

**Algorithme 3** Exemple : calcul des itemsets fréquents

Items I = {A, B, C, D, E, F}
Transactions T = {AB, ABCD, ABD, ABDF, ACDE, BCDF}
MINSUP = 1/2
Calcul de L1 (ensemble des 1-itemsets) :
C1 = I = {A,B,C,D,E,F} // C1 : ensemble de 1-itemsets candidats
support(A) = support(B) = 5/6, support(C) = 3/6, support(D) = 5/6, support(E) = 1/6, support(F) = 2/6
L1 = {A, B, C, D}
Calcul de L2 (ensemble des 2-itemsets) :
C2 = L1xL1 = {AB,AC, AD, BC, BD, CD}
support(AB) = 4/6, support(AC) = 2/6, support(AD) = 4/6, support(BC) = 2/6, support(BD) = 4/6, support(CD) = 3/6
L2 = {AB,AD, BD, CD}
Calcul de L3 (ensemble des 3-itemsets) :
C3 = {ABD} (ABC $\not\subseteq$ C3 car AC $\not\subseteq$ L2)
support(ABD) = 3/6 L3 = {ABD}
Calcul de L4 (ensemble des 4-itemsets) :
C4 =$\phi$
L4 =$\phi$
Calcul de L (ensembles des itemsets fréquents) :
L = $\cup_i L_i$ = {A, B, C, D, AB, AD, BD, CD, ABD}

colonnes, où les lignes sont les transactions de la BD et les colonnes sont les différentes valeurs possibles de l'attribut indexé. Si les attributs sont continus alors la construction de l'index est plus difficile.

**Alternatives :**
Dans une logique d'exploration du treillis en profondeur et non plus en largeur, d'autres algorithmes ont été proposés, tel que FP-Growth [HPY00], Éclat [ZPOL97] ou Close [NPL99].

*FP-Growth* utilise une représentation très condensée des données pour évaluer les fréquences par comptage dans la base.

*Eclat* recherche en profondeur dans le treillis par intersection rapide des tid-listes, la procédure étant interrompue dès que l'on est sûr que l'itemset candidat ne peut plus être fréquent.

*Close* extrait les itemsets fermés fréquents, qui constituent une partie génératrice des itemsets fréquents et de leur support, ce qui réduit les temps d'extraction et produit des règles non redondantes.

*MaxEclat* [ZPOL97] ou *Max-Miner* [RB98] sont des algorithmes de recherche d'itemsets fréquents maximaux où les sur-ensembles d'itemsets sont non-fréquents. Mais ils se prêtent mal au calcul du support des ensembles fréquents qu'ils contiennent, calcul pourtant nécessaire pour le calcul de la confiance des règles générées.

#### 3.3.2.3 Versions parallèles

Il existe de nombreux algorithmes parallèles et distribués basés sur la recherche d'itemsets fréquents et notamment sur l'algorithme Apriori. Dans [Zak99], Zaki propose de classer les algorithmes par stratégie d'équilibrage de charge (load-balancing), architecture et parallélisme. Le parallélisme apporte deux perspectives intéressantes : parallélisme de données et parallélisme de tâches [CDG+97]. Les deux paradigmes diffèrent par le fait que l'ensemble de candidats soit distribué ou non à travers les processeurs. Dans le paradigme de parallélisme de données, chaque nœud compte le même ensemble de candidats alors que dans le paradigme de parallélisme de tâches, l'ensemble des candidats est divisé et distribué à travers les processeurs, et chaque nœud compte un ensemble différent de candidats. La base de données peut théoriquement être divisée dans l'un ou l'autre paradigme. Dans la pratique, pour un I/O plus efficace on suppose habituellement que la base de données est divisée et distribuée à travers les processeurs.

**Parallélisme de données :**
Les algorithmes qui adoptent le paradigme de parallélisme de données incluent : Count Distribution [AS96], PDM (Parallel Data Mining) [PCY95], DMA (Distributed Mining Algorithm) [CNFF96], et CCPD (Common Candidate Partitioned Database) [ZOPL96]. Ces algorithmes parallèles diffèrent par l'utilisation ou non de techniques d'élimination de candidats (pruning) ou de technique de comptage de candidats efficaces.

**Parallélisme de tâches :**
Les algorithmes adoptant le paradigme de parallélisme de tâche incluent : DD (Data Distribution) [AS96], IDD (Intelligent Data Distribution) [HTK97], HPA (Hash-based Parallel mining of Association rules) [SK96] et PAR (Parallel Association Rules) [ZPOL97], qui inclue les algorithmes (Par-Eclat, Par-MaxEclat, Par-Clique et Par-MaxClique). Ils divisent tous aussi bien les candidats que la base de données parmi les processeurs. Ils diffèrent dans la façon dont les candidats et la base de données sont divisés.

## 3.4 Mesures de qualité existantes

Plusieurs mesures de qualité des règles ont été proposées par différentes communautés telles que les statistiques, le datamining ou l'optimisation. Nous présentons ici quelques unes des principales mesures.

Notation : Soit une règle d'association R de la forme C $\longrightarrow$P (IF C THEN P), dans la suite du chapitre on désigne par :

- N : le nombre total d'instances dans la base.
- |P| : le nombre d'instances de la base de données contenant la partie P de la règle.
- $|\overline{P}|$ : le nombre d'instances de la base de données ne contenant pas la partie P de la règle.
- |C et P| : le nombre d'instances de la base de données contenant à la fois la partie C et la partie P.
- $Pr$(C) : la probabilité de C.

### Le Support et la Confiance

La qualité d'une règle est souvent mesurée par deux indicateurs principaux : le *support* et la *confiance*. Une règle est plus ou moins vraie pour un certain pourcentage d'instances constituant le support de la règle. Le *support* est défini par la probabilité $Pr$(C, P). On a :

$$Support(R) = \frac{|CetP|}{N} \tag{3.1}$$

Le *support* n'est pas suffisant pour évaluer la qualité d'une règle. Il se peut en effet qu'une règle corresponde à des items (objets) apparaissant rarement, mais soit très valide.

La *confiance* correspond à la probabilité conditionnelle $Pr$(P sachant C) qui est la probabilité a posteriori que P soit vraie étant donné que C est vraie. On a :

$$Confiance(R) = \frac{|CetP|}{|C|} \tag{3.2}$$

La mesure de ***laplace*** est une autre mesure équivalente à la *confiance* qui est généralement utilisée pour mesurer la précision des règles de classification [CB91]. La mesure *laplace* donne une valeur positive non nulle et inférieure à 1, elle est définie comme suit :

$$Laplace(R) = \frac{|CetP| + 1}{|C| + 2} \tag{3.3}$$

Si C et P sont statistiquement indépendants alors la *confiance* de la règle R vaut 0, et la valeur de *laplace* appartient à l'intervalle ]0 ,0.5]. La *confiance* ne suffit pas pour caractériser la dépendance de C et P, car une forte *confiance* peut être liée à une forte probabilité de P. Ainsi, Brain et al. [BMUT97][BMS97] suggèrent d'autres mesures de dépendance comme le *khi-deux* et l'*intérêt*.

### KHI-DEUX $\chi^2$

C'est une mesure de dépendance qui vient de la statistique [Gue73]. Deux variables nominales C et P sont indépendantes si et seulement si $Pr$(C, P)=$Pr$(C)*$Pr$(P).

$$\chi^2(R) = \frac{N * (|CetP| * |\overline{C}et\overline{P}| - |Cet\overline{P}| * |\overline{C}etP|)^2}{|C| * |P| * |\overline{C}| * |\overline{P}|} \tag{3.4}$$

Plus la valeur de $\chi^2$ augmente, plus l'indépendance est faible et plus la corrélation est élevée. Le $\chi^2$ est sensible à la valeur de N.

### Phi-coefficient $\phi$

C'est une mesure de dépendance linéaire dérivée du test de $\chi^2$ ($\phi^2 = \frac{\chi^2}{N}$), elle est équivalente à la corrélation de Pearson $r$ [LER81].

- C et P sont indépendantes si $\phi(R) = 0$.
- C et P sont corrélées positivement si $\phi(R) \in ]0, 1]$
- C et P sont corrélées négativement si $\phi(R) \in [-1, 0[$
- Un *support* fort implique un Phi-coefficient fort.

### Intérêt et Conviction

L'*intérêt* (appelé aussi $lift$) mesure la dépendance en privilégiant les motifs rares (dont le *support* est faible). Cette mesure est très utilisée par la communauté datamining. Plus la valeur s'éloigne de 1, plus C et P sont dépendants.

$$Intérêt(R) = \frac{N * |CetP|}{|C| * |P|} = \frac{Confiance(R)}{Pr(P)} \tag{3.5}$$

L'*intérêt* de la règle est intéressant dans la région de faible *support*. Il prend des valeurs dans l'intervalle [0 , $\infty$[.

- C et P sont indépendantes si Intérêt(R)=1.
- La règle est intéressante si Intérêt(R)$\in$ ]1 $\infty$[

Cependant, l'*intérêt* a un comportement symétrique. Afin de pallier à ce problème, Brin et al. [BMUT97] ont défini un nouvel indice : la *conviction*.

$$Conviction(R) = \frac{|C| * |\overline{P}|}{N * |Cet\overline{P}|} = \frac{Pr(\overline{P})}{Confiance(C \longrightarrow \overline{P})} \tag{3.6}$$

Celle-ci mesure la faiblesse de la règle C$\longrightarrow\overline{P}$. Elle est symétrique en complément. Plus la *conviction* est élevée, plus la dépendance est élevée.

### Cosinus

Cette mesure est également dérivée de la corrélation statistique, elle est très intéressante dans la région de faible *support* et de fort *intérêt*, elle est fortement corrélée avec le *phi-coefficient* dans ces régions.

$$Cosinus(R) = \frac{|CetP|}{\sqrt{|C| * |P|}} \tag{3.7}$$

Cette mesure agrège différentes informations car :

- Elle peut s'exprimer à partir du *support* et de l'*intérêt* (donc elle regroupe ces deux mesures) :

$$Cosinus(R) = \sqrt{Intérêt(R) * Support(R)} \tag{3.8}$$

- C'est la moyenne géométrique de la *confiance* des deux règles $C \longrightarrow P$ et $P \longrightarrow C$ :

$$Cosinus(R) = \sqrt{Confiance(C \longrightarrow P) * Confiance(P \longrightarrow C)} \tag{3.9}$$

### Surprise

Elle est utilisée pour mesurer l'affirmation. Elle permet de chercher les règles étonnantes. Moins P est répandue, plus il est étonnant de trouver une bonne confirmation de la règle. Plus le *support* est fort et la fréquence de $\overline{P}$ est faible, plus la *surprise* est forte (la règle est plus étonnante) [AK01].

$$Surprise(R) = \frac{|CetP| - |Cet\overline{P}|}{|\overline{P}|} \tag{3.10}$$

La *surprise* prend des valeurs réelles positives ou nulles (Surprise(R)$\in$ [0 , $\infty$[).

### Fiabilité

Elle mesure l'effet (impact) de la disponibilité de l'information de C dans la probabilité de P. Une grande *Fiabilité* implique une forte association entre C et P [LFZ99].

$$Fiabilite(R) = \frac{|CetP|}{|C|} - \frac{|P|}{N} \tag{3.11}$$

- C et P sont indépendantes si $Fiabilité(R) = 0$.
- C et P sont corrélées positivement si $Fiabilité(R) \in ]0, 1]$
- C et P sont corrélées négativement si $Fiabilité(R) \in [-1, 0[$

### Piatetsky-Shapiro

C'est une mesure de dépendance proposée par Piatetsky-Shapiro [PS91]. Elle est utilisée pour quantifier la corrélation entre deux attributs dans une classification simple.

$$Piatetsky\_Shapiro(R) = \frac{|CetP|}{N} - \frac{|C|}{N} * \frac{|P|}{N} \tag{3.12}$$

- si $Piatetsky\_Shapiro(R) = 0$ alors C et P sont indépendantes.
- si $Piatetsky\_Shapiro(R) \in ]0, 0.25]$ alors C et P sont corrélées positivement.
- si $Piatetsky\_Shapiro(R) \in [-0.25, 0[$ alors C et P sont corrélées négativement.

### J-mesure

Elle est proposée par Goodman et Smith [GS88] et est utilisée en optimisation combinatoire. La *J-mesure* est donnée par la formule suivante :

$$J - masure(R) = Pr(P) * [Pr(C/P) \log(\frac{Pr(C/P)}{Pr(C)}) + (1 - Pr(C/P)) \log(\frac{1 - Pr(C/P)}{1 - Pr(C)})] \tag{3.13}$$

Le premier terme *Pr(C)* est considéré comme la préférence de généralité ou la simplicité de la règle, le reste représente l'entropie relative (similarité) [GS91].
Une grande valeur de J-measure n'est pas nécessairement associée à une bonne règle (dans le cas où la *confiance* est faible et le *support* est fort) [HH99][HV02]. Une solution consiste à supprimer le deuxième terme. La formule devient alors :

$$J1 - mesure(R) = Pr(P) * [Pr(C/P) \log(\frac{Pr(C/P)}{Pr(C)})] \tag{3.14}$$

Cette formule pose également un problème pour les algorithmes évolutionnaires. En effet, dans les premières générations, la valeur de |C| est faible. La raison est que, C étant généré aléatoirement, il est peu évident qu'il soit en adéquation avec les instances de la base de données. Les individus des premières générations ont alors une faible mesure, ce qui rend possible la situation où aucun individu n'est sélectionné. Pour pallier à ce problème, Freitas [Fre99] a proposé une amélioration de la formule de la manière suivante :

$$J'mesure(R) = \frac{\omega 1 * (J1 - mesure) + \omega 2 * (\frac{Npu}{NT})}{\omega 1 + \omega 2} \tag{3.15}$$

L'expression de $J1 - mesure$ a été définie précédemment. $Npu$ représente le nombre d'attributs potentiellement utiles dans la partie C de la règle. Un attribut est potentiellement utile s'il apparaît dans la partie C d'au moins un individu. $NT$ est le nombre total d'attributs apparaissant dans la partie C de la règle. $\omega 1$ et $\omega 2$ sont deux paramètres choisis par l'utilisateur, compris entre 0 et 1 et dont la somme est égale à 1.

### Jaccard

C'est une mesure de similarité, essentiellement utilisée pour calculer la distance entre deux mots.

$$Jaccard(R) = \frac{|CetP|}{|C| + |P| - |CetP|} \tag{3.16}$$

### Rappel

Elle permet d'évaluer la proportion des transactions vérifiant la partie $C$ de la règle parmi celles vérifiant la partie conclusion $P$ de la règle $C \longrightarrow P$ [LFZ99].

$$Rappel(R) = \frac{|CetP|}{|P|} \tag{3.17}$$

### Pearl

La mesure *Pearl* permet de mesurer l'intérêt de la règle par rapport à l'hypothèse d'indépendance entre la partie condition et conclusion de la règle [Pea88].

$$Pearl(R) = \frac{|C|}{N} * \frac{|CetP|}{|C|} - \frac{|P|}{N} \tag{3.18}$$

### Loevinger

C'est une ancienne mesure de qualité citée dans le domaine de datamining (1947). Elle introduit la probabilité des transactions qui ne vérifient pas la conclusion. Comme la mesure *conviction*, elle permet donc de pallier à l'un des défauts de la *confiance* [Loe47].

$$Loevinger(R) = \frac{\frac{|CetP|}{|C|} - \frac{|P|}{N}}{\frac{|\overline{P}|}{N}} \tag{3.19}$$

### Sebag

Cette mesure prend en compte de manière explicite le nombre de contre-exemples. Elle calcule simplement le rapport entre le nombre d'exemples de la règle et son nombre de contre-exemples. Dès que la mesure est supérieure à 1 alors la règle possède plus d'exemples que de contre-exemples et inversement, dès que la mesure est inférieure à 1 alors la règle est plus souvent infirmée par les données plutôt que confirmée [SS88].

$$Sebag(R) = \frac{|CetP|}{|Cet\overline{P}|} \tag{3.20}$$

### Satisfaction

Cette mesure peut se réécrire simplement en $1 - Interet(C \longrightarrow \overline{P})$. Donc, c'est l'inverse de l'*Interet*. La satisfaction permet donc d'apprécier si la règle $C \longrightarrow P$ est plus intéressante que la règle $C \longrightarrow \overline{P}$. Lorsque le nombre de contre-exemples de la règle $C \longrightarrow P$ augmente alors le nombre de contre-exemples de la règle $C \longrightarrow \overline{P}$ augmente et la satisfaction de $C \longrightarrow P$ diminue [LFZ99].

$$Satisfaction(R) = \frac{\frac{|\overline{P}|}{N} - \frac{|Cet\overline{P}|}{|C|} - \frac{|P|}{N}}{\frac{|\overline{P}|}{N}} \tag{3.21}$$

### Spécificité

La spécificité représente la *confiance* de la règle $\overline{P} \longrightarrow \overline{C}$. Cette règle possède les mêmes contre-exemples que la règle $C \longrightarrow P$. La règle $C \longrightarrow P$ ayant une spécificité élevée indique que la probabilité d'observer l'attribut $\overline{C}$ est élevée sachant que l'on a observé l'attribut $\overline{P}$. Donc, le nombre de transactions vérifiant $\overline{C}$ et $\overline{P}$ est faible ce qui confirme l'*Interet* pour la règle $C \longrightarrow P$ qui possède alors elle aussi peu de contre-exemples [LFZ99].

$$Spécificité(R) = \frac{|\overline{C} et \overline{P}|}{|\overline{P}|} \tag{3.22}$$

### Autres mesures

En plus des mesures présentées auparavant, il existe d'autres mesures telles que : Gini Index, Collective strength, Certainty factor, Klosgen, Odds ratio, Yules'Q, Yules'Y, Kappa, Mutual Information, Added value, goodman-Krustal's, Agrawal and Srikant's Itemset Measure, Itemset Klemettinen et al. Rule Templates, Gray and Orlowska's Interestingness, Liu et al. Reliable Exceptions, Zhong et al. Peculiarity.

Pour plus de détails sur ces mesures le lecteur peut se référer à [TKS02][HH99][LFZ99].

## 3.5 Propriétés des bonnes mesures

Parmis toutes les mesures exposées précédemment, une question fondamentale est quelle mesure utiliser. Dans une approche monocritère, une voire deux mesures au plus peuvent être considérées. Il est alors nécessaire d'étudier les propriétés de ces mesures. Plusieurs études ont été menées dans ce sens.
Nous exposons ici les travaux de la littérature à ce sujet et présenterons dans la partie suivante notre approche.

### 3.5.1 Première approche : Propriétés probabilistes

#### Principes de caractérisation

V. Shi et al. [SDP01] ont proposé six principes pour caractériser une bonne mesure de qualité d'une règle d'association : l'implication, la corrélation, la nouveauté, l'utilité, le top-N-règle et l'efficacité.

- *Propriété SH1* (Implication) : une mesure $M$ satisfait le principe d'implication si et seulement si $M(C \longrightarrow P) > M(P \longrightarrow C) si Pr(C) < Pr(P)$.
- *Propriété SH2* (Corrélation) : une mesure satisfait le principe de corrélation si elle est proportionnelle à la covariance de C et P.
- *Propriété SH3* (Nouveauté) : une mesure satisfait le principe de nouveauté si elle est inversement proportionnelle au maximum des deux probabilités $Pr$(C) et $Pr$(P).
- *Propriété SH4* (Utilité) : une mesure $M$ satisfait le principe d'utilité si $M$ est une fonction croissante et monotone avec $Pr$(C, P) et vérifie un certain seuil (règle fréquente).
- *Propriété SH5* (Top-N-règle) : une mesure satisfait le principe de Top-N-règle si la mesure permet de trier les règles pour générer N meilleures règles afin que l'utilisateur puisse faire son choix.
- *Propriété SH6* (Efficacité) : c'est l'efficacité en temps de calcul.

Les deux derniers principes sont liées à l'utilisateur qui décide quelle est la meilleure mesure en regardant aussi les autres principes. Le tableau 3.1 présente les propriétés de cinq mesures (*support, confiance, Intérêt, Conviction, Fiabilité)* telles que l'ont présenté Shi et al. [SDP01].

| | *Support* | *Confiance* | *Intérêt* | *Conviction* | *Fiabilité* |
|---|---|---|---|---|---|
| Implication | | X | | X si corrélation positive | X si corrélation négative |
| Corrélation | | | X | X | X |
| Nouveauté | | | X | | X si corrélation négative |
| Utilité | X | | | | |

TAB. 3.1 – Propriétés de cinq mesures classiques.

### Propriétés des mesures

D'autre part, Piatesky-Shapiro [PS91] a proposé trois propriétés essentielles devant être vérifiées par une bonne mesure $M$ :

- **Propriété PS1** : $M = 0$ si C et P sont indépendants.
- **Propriété PS2** : $M$ est monotone croissante en fonction de la probabilité Pr(C,P) si Pr(C) et Pr(P) restent les mêmes.
- **Propriété PS3** : $M$ est monotone décroissante en fonction de la probabilité Pr(C) (resp. Pr(P)) si les autres paramètres Pr(C,P) et Pr(P) (resp. Pr(C)) restent inchangés.

Tan et al. [TKS02] et Hilderman et al. [HH01] ont étendu ces trois propriétés en cinq autres propriétés en introduisant la symétrie et les différentes permutations entres les variables (P et C) et entre les colonnes et les lignes de la table de contingence. Soit la formulation matricielle : chaque table de contingence est représentée par une matrice de contingence $MC = \begin{pmatrix} f11 & f10 \\ f10 & f00 \end{pmatrix} = \begin{pmatrix} |CandP| & |Cand\overline{P}| \\ |\overline{C}andP| & |\overline{C}and\overline{P}| \end{pmatrix}$ et les mesures opèrent sur cette matrice de contingence.
Si le déterminant de cette matrice est nul ($Det(MC) = 0$), l'indépendance statistique est vérifiée.

**Propriété P4** : La symétrie.
Une mesure $M$ est symétrique si $M(C \longrightarrow P) = M(P \longrightarrow C)$, ce qui est équivalant dans la formulation matricielle à $M(MC) = M(Transpose(MC))$, sinon elle est non symétrique (asymétrique). En pratique les mesures asymétriques sont utilisées pour les règles d'implication afin de faire la différence entre les deux règles $C \longrightarrow P$ et $P \longrightarrow C$.
Exemple des mesures asymétriques : *laplace, confiance, conviction, J-mesure.*
*Exemple des mesures symétriques :* intérêt, *$\phi$-coefficient*, Cosinus.

**Propriété P5** : Invariance sous Ligne/colonne scaling.
Soit Lg et Cl deux matrices avec $Lg = Cl = \begin{pmatrix} k1 & 0 \\ 0 & k2 \end{pmatrix}$ où $k1$ et $k2$ sont deux constantes positives.
Le produit Lg*MC revient à multiplier (scaling) la première ligne (fréquence de C) par $k1$ et la seconde ligne (fréquence de $\overline{C}$) par $k2$, et le produit MC*Cl correspond à multiplier (scaling) la première colonne (fréquence de P) par $k1$ et la seconde colonne (fréquence de $\overline{P}$) par $k2$.
La mesure M satisfait cette propriété si $M(MC) = M(Lg * MC) = M(MC * Cl)$.
Il y a trois mesures qui vérifient cette propriété : Odd ratio, Yule's Q et Yule's Y.

**Propriété P6** : Anti-symétrie sous Ligne/colonne permutation.
Soit S une matrice de permutation : $S = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Une mesure normalisée M est anti-symétrique sous permutation des lignes (la règle $\overline{C} \longrightarrow P$) si $M(S * MC) = -M(MC)$ et anti-symétrique sous permutation des colonnes (la règle $C \longrightarrow \overline{P}$) si $M(MC * S) = -M(MC)$.
Le $\phi$-coefficient, Piatesky Shapiro's, Yule's Q et Yule's Q sont des exemples de mesures vérifiant cette propriété.

**Propriété P7** : Inversion Invariance.
Soit S une matrice de permutation $S = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
Cette propriété est un cas particulier des propriétés précédentes, on fait les deux permutations des lignes et des colonnes à la fois. Une mesure normalisée M vérifie l'invariance sous inversion si $M(S * MC * S) = M(MC)$.
Elle permet de distinguer les mesures binaires symétriques des mesures binaires asymétriques.

**Propriété P8** : Null-Invariance.
Soit CC la matrice suivante : $CC = \begin{pmatrix} 0 & 0 \\ 0 & k \end{pmatrix}$ où $k$ est une constante positive.
Une mesure binaire M est Null-Invariance si $M(MC + CC) = M(MC)$.
Cette propriété vérifie la stabilité de la mesure lorsque l'on rajoute des instances qui ne contiennent pas C ou P.

Cette approche probabiliste permet de mettre en évidence quelques propriétés sur les mesures. Ces propriétés peuvent être complétées avec d'autres approches d'analyse.

### 3.5.2 Approche fonctionnelle

Lallich et Teytaud [LT03] ont étudié les mesures de qualité sous une vue fonctionnelle. Plusieurs mesures s'écrivent comme une normalisation de la *confiance* par le biais d'une transformation affine de la mesure : $m(C \longrightarrow P) = \theta_1(Confiance(C \longrightarrow P) - \theta_2)$, dont

les paramètres ne dépendent que des marges relatives de la table de contingence qui croise $C$ et $P$ et éventuellement du nombre de transactions $N$. Le plus souvent, tout revient à un centrage-réduction, où le changement d'origine amène à comparer $Pr(P \setminus C)$ à $Pr(P)$, sa valeur attendue en l'absence de liaison, alors que le changement d'échelle varie suivant le but poursuivi. A l'inverse, la lecture des changements d'échelle permet de savoir ce qui différencie deux mesures centrées sur $Pr(P)$. Le tableau 3.2 présente les deux paramètres $\theta_1$ et $\theta_2$.

| Mesure | $\theta_1$ | $\theta_2$ |
|---|---|---|
| Intérêt | 0 | $\frac{1}{Pr(P)}$ |
| Loevinger | $Pr(P)$ | $\frac{1}{Pr(\overline{P})}$ |
| Piatetsky Shapiro | $Pr(P)$ | $Pr(C)$ |
| Pearl | $Pr(P)$ | $Pr(C)$ |
| $\phi$ correlation | $Pr(P)$ | $\frac{\sqrt{Pr(C)}}{\sqrt{Pr(\overline{C})*Pr(P)*Pr(\overline{P})}}$ |

TAB. 3.2 – Transformation affine de la mesure : $m(C \longrightarrow P) = \theta_1(Confiance(C \longrightarrow P) - \theta_2)$.

L'étude des mesures de qualité avec cette voie permet de mettre en évidence des relations lors de la présentation des mesures, mais ne peut pas être étendue à toutes les mesures.

### 3.5.3 Troisième approche : Préférences utilisateur (expert)

Cette approche vise à rationaliser le processus de décision d'un expert dans le choix de la mesure à utiliser. Comment prendre en compte les préférences d'un utilisateur dans la définition de critères d'évaluation de mesures. Ces critères, qui peuvent prendre des valeurs aussi bien numériques que symboliques, sont ensuite traduits sur des échelles de préférences propres à chaque utilisateur. Ces critères font ensuite l'objet de traitement par des algorithmes d'aide à la décision [PL03].
Une règle est jugée intéressante selon la mesure $M$ lorsque $M(C \longrightarrow P) \geq \alpha$. Une famille de critères $(g_i)$ permettent de mesurer les préférences d'un utilisateur (tableau 3.3).

Les trois critères $g_{12}$, $g_3$ et $g_4$ permettent de construire une matrice de décision (mesure $\times$critère). La matrice de décision est remplie par les évaluations de chaque mesure sur chaque critère $(g_i)$. La sélection de bonnes mesures se fait à partir des trois méthodes d'aide à la décision : la somme pondérée, ELECTRE I [Roy85] et PROMETHEE [BM94].

Cette approche vise donc à aider un décideur lors de choix de sa mesure. Trois propriétés sont étudiées à cette fin : Indépendance, Linéarité et Symétrie.

Nous avons évoqué précédemment les propriétés et préférences pour mesurer le qualité de règles d'association proposées par différents auteurs. Ces travaux apportant un éclairage probabiliste et théorique pour caractériser une bonne mesure de qualité d'une règle d'association.

| $g$ | Sémantique | Valeur | | |
|---|---|---|---|---|
| $g_1$ | Facilité à placer le seuil $\alpha$ par rapport à la valeur maximale prise par la mesure | 0 (en fonction des paramètres)<br>1 (infini), 2(fixe) | | |
| $g_2$ | Facilité à placer le seuil $\alpha$ par rapport à l'hypothèse d'indépendance | 0 (en fonction des paramètres)<br>1 (fixe) | | |
| $g_3$ | Linéarité de la mesure | 0 (linéaire)<br>1 (non linéaire) | | |
| $g_4$ | Symétrie de la mesure | 0 (symétrique)<br>1 (non symétrique) | | |
| $g_{12}$ | (à partir de $g_1$ et $g_2$)<br>Facilité à placer le seuil $\alpha$ par rapport à la valeur maximale prise par la mesure et la valeur d'indépendance | **Indé. ($g_2$)** | **borne ($g_1$)** | **valeur** |
| | | param. | param. | 0 |
| | | param. | $\infty$ | 1 |
| | | param. | fixe | 2 |
| | | fixe | param. | 3 |
| | | fixe | $\infty$ | 4 |
| | | fixe | fixe | 5 |

TAB. 3.3 – Famille de critères.

Dans la section suivante, nous allons proposer une approche statistique. Le but de cette étude est de mettre en évidence les corrélations existantes entre les mesures afin de regrouper les critères ayant le même comportement (et mesurant donc les mêmes propriétés) et de déterminer un ensemble cohérent de critères complémentaires pour une modélisation multi-objectif du problème.

## 3.6 Notre approche : Analyse statistique des critères

Notre approche est différente des approches proposées dans la littérature. En effet, nous avons cherché à étudier les critères en fonction des résultats qu'ils produisaient. Le but de cette étude est de mettre en évidence les corrélations existantes entre les mesures afin de regrouper les critères ayant le même comportement (et mesurant donc les mêmes propriétés) et déterminer un ensemble cohérent de critères complémentaires.

Afin d'étudier les relations entre critères, nous avons réalisé une analyse statistique pour des bases de données classiquement utilisées en data mining *UCI Machine Learning Repository* [NHBM98] et pour des bases relatives à des expérimentations sur puces à ADN (voir plus loin 5.4), les résultats sont semblables. Nous allons présenter dans le reste du chapitre les résultats trouvés pour la base de données génomique "MIPS Yeast Genome Database" (une base de données contenant 2467 gènes pour 79 puces).

Ainsi, pour chaque problème, nous avons énuméré toutes les règles pouvant exister (énumération exhaustive). Nous avons alors mesuré chacune de ces règles suivant un ensemble de 24 mesures (voir tableau 3.4) : Support (Sup), Confiance (Cof), Intérêt (Int), Conviction (Conv), Surprise (Sur), Jaccard (Jac), Phi-coefficient (Phi), Cosinus (Cos), Jmesure (Jme), Piatesky-Shapiro (Pia), Laplace (Lap), Spécificité (Spe), Pearl (Pea), Fia-

bilité (fia), Satisfaction (Sat), Sebag (Seb), Collective (Col), Rappel (Rap), Klosgen (Klo), Loevinger (Loe), Kappa (Kapa), Odds (Odd), Qyules (YuQ) et Yyules (YuY).

Nous avons ainsi généré avec une énumération exhaustive un tableau de données dans lequel une ligne représente une règle d'association et chaque colonne indique la qualité de la règle par rapport à l'une des mesures étudiées (24 colonnes).

Deux types d'analyse statistique ont été faites. Une première étude concerne l'analyse descriptive univariée de chaque mesure par l'utilisation des outils statistiques du logiciel SPSS version 10. Une deuxième analyse descriptive multivariée de type ACP (Analyse en Composantes Principales) complète cette première étude exploratoire.

### 3.6.1 Analyse descriptive univariée

Le tableau 3.6 ci-après, décrit l'histogramme de chaque mesure observé sur l'ensemble des règles ainsi que l'allure (courbe gaussienne) de ces mesures. Nous donnons dans le tableau 3.5 : la moyenne (Moy) ainsi que les estimations des coefficients d'asymétrie (g1) et d'aplatissement (g2) obtenus à l'aide de SPSS [Inc03][Bli67][Spi72], pour apprécier et comparer les mesures d'un point de vue des courbes de densité et de leur approximation par la loi normale.

A partir de ces graphiques et des coefficients $g1$, il apparaît que les mesures *Surprise, Sebag, Intérêt, Convition et Odds* ont une dissymétrie gauche beaucoup plus prononcée que *support* (par exemple g1=4,484 pour la Surprise et g1=1,459 pour le *support*) avec quelques valeurs très éloignées de la moyenne (g2 = 33,638 pour la Surprise ). Les mesures *Satisfaction et Loevinger* ont, à l'inverse, une dissymétrie droite avec quelques valeurs très éloignées de la moyenne.

Les mesures *Cosinus, Jaccard, Rappel et Pearl* ont une faible dissymétrie gauche.

Globalement l'allure des histogrammes et des valeurs de g1 et de g2 permettent de rapprocher les mesures de *Piatesky-Shapiro*, *J-mesure*, *laplace*, *confiance*, *$\phi$-coefficient*, *Collective*, *Klosgen*, *Fiabilité*, *Spécificité*, *Kappa*, *Yule's Y* et *Yule's Q* à la distribution normale. Ensuite viennent *Cosinus*, *Jaccard*, *Rappel*, *Pearl* et *support* qui tendent vers une forme similaire. Alors que *Intérêt*, *Conviction*, *Sebag*, *Odds* et *Surprise* possèdent une distribution très irrégulière et ont une répartition opposée à *Satisfaction* et *Loevinger*.

### 3.6.2 Analyse descriptive multivariée : ACP

Nous avons ensuite soumis le tableau de données où les lignes sont les règles et les colonnes leur évaluation suivant les différents critères, à l'analyse en composantes principales normée [LMP95] disponible sous le logiciel SPAD 5.5 [DEC02]. Des corrélations fortes entre mesures revient donc à trouver des corrélations entre les colonnes de la matrice. Ainsi on peut mettre en évidence des critères ayant des comportements similaires pour l'ensemble des règles. L'ensemble de ces comportements est résumé par la figure 3.4 (matrice des corrélations linéaires entre les 24 critères).

| Mesure | Formule |
|---|---|
| Support | $\frac{\lvert CetP\rvert}{N}$ |
| Confiance | $\frac{\lvert CetP\rvert}{\lvert C\rvert}$ |
| Laplace | $\frac{\lvert CetP\rvert+1}{\lvert C\rvert+2}$ |
| $\chi^2$ | $\frac{N*(\lvert CetP\rvert*\lvert \overline{CetP}\rvert-\lvert CetP\rvert*\lvert \overline{CetP}\rvert)^2}{\lvert C\rvert*\lvert P\rvert*\lvert \overline{C}\rvert*\lvert \overline{P}\rvert}$ |
| $\phi^2$ | $\frac{\chi^2}{N}$ |
| Intérêt | $\frac{N*\lvert CetP\rvert}{\lvert C\rvert*\lvert P\rvert}$ |
| Conviction | $\frac{\lvert C\rvert*\lvert \overline{P}\rvert}{N*\lvert Cet\overline{P}\rvert}$ |
| Cosinus | $\frac{\lvert CetP\rvert}{\sqrt{\lvert C\rvert*\lvert P\rvert}}$ |
| Surprise | $\frac{\lvert CetP\rvert-\lvert Cet\overline{P}\rvert}{\lvert \overline{P}\rvert}$ |
| Fiabilité | $\frac{\lvert CetP\rvert}{\lvert C\rvert}-\frac{\lvert P\rvert}{N}$ |
| Piatesky-Shapiro | $\frac{\lvert CetP\rvert}{N}-\frac{\lvert C\rvert}{N}*\frac{\lvert P\rvert}{N}$ |
| Jmeasure | $\frac{\lvert P\rvert}{N}*[\frac{\lvert CetP\rvert}{\lvert P\rvert}\log(\frac{\frac{\lvert CetP\rvert}{\lvert P\rvert}}{\frac{\lvert C\rvert}{N}})$ $+(1-\frac{\lvert CetP\rvert}{\lvert P\rvert})\log(\frac{1-\frac{\lvert CetP\rvert}{\lvert P\rvert}}{1-\frac{\lvert C\rvert}{N}})]$ |
| Jaccard | $\frac{\lvert CetP\rvert}{\lvert C\rvert+\lvert P\rvert-\lvert CetP\rvert}$ |
| Rappel | $\frac{\lvert CetP\rvert}{\lvert P\rvert}$ |
| Pearl | $\frac{\lvert C\rvert}{N}*\frac{\lvert CetP\rvert}{\lvert C\rvert}-\frac{\lvert P\rvert}{N}$ |
| Loevinger | $\frac{\frac{\lvert CetP\rvert}{\lvert C\rvert}-\frac{\lvert P\rvert}{N}}{\frac{\lvert \overline{P}\rvert}{N}}$ |
| Sebag | $\frac{\lvert CetP\rvert}{\lvert Cet\overline{P}\rvert}$ |
| Satisfaction | $\frac{\frac{\lvert \overline{P}\rvert}{N}-\frac{\lvert Cet\overline{P}\rvert}{\lvert C\rvert}-\frac{\lvert P\rvert}{N}}{\frac{\lvert \overline{P}\rvert}{N}}$ |
| Spécificité | $\frac{\lvert \overline{CetP}\rvert}{\lvert \overline{P}\rvert}$ |
| Kappa | $\frac{N(\lvert CetP\rvert+\lvert \overline{CetP}\rvert)-\lvert C\rvert*\lvert P\rvert-\lvert \overline{C}\rvert*\lvert \overline{P}\rvert}{N*N-\lvert C\rvert*\lvert P\rvert-\lvert \overline{C}\rvert*\lvert \overline{P}\rvert}$ |
| Collective strength | $N*\frac{(\lvert CetP\rvert+\lvert \overline{CetP}\rvert)}{\lvert C\rvert*\lvert P\rvert+\lvert \overline{C}\rvert*\lvert \overline{P}\rvert}*\frac{N*N-\lvert C\rvert*\lvert P\rvert-\lvert \overline{C}\rvert*\lvert \overline{P}\rvert}{N*(N-\lvert CetP\rvert-\lvert \overline{CetP}\rvert)}$ |
| Klosgen | $\sqrt{\frac{\lvert CetP\rvert}{N}}(\frac{\lvert CetP\rvert}{\lvert C\rvert}-\frac{\lvert P\rvert}{N})$ |
| Odds ratio ($\alpha$) | $\frac{(\lvert CetP\rvert*\lvert \overline{CetP}\rvert)}{(\lvert Cet\overline{P}\rvert*\lvert \overline{CetP}\rvert)}$ |
| Yules'Q | $\frac{\alpha-1}{\alpha+1}$ |
| Yules'Y | $\frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$ |

TAB. 3.4 – Mesures étudiées.

| | support | confiance | interet | conviction | surprise | jaccard | coefficient | cosine | jmeasure | piatetsky | laplace | specificite | sebag | fiabilite | satisfaction | pearl | rappel | collective | klosgen | Loevinger | kappa | odds | Qyules | Yyules |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **support** | 1,00 | | | | | | | | | | | | | | | | | | | | | | | |
| **confiance** | 0,41 | 1,00 | | | | | | | | | | | | | | | | | | | | | | |
| **interet** | -0,01 | 0,12 | 1,00 | | | | | | | | | | | | | | | | | | | | | |
| **conviction** | 0,28 | 0,59 | 0,36 | 1,00 | | | | | | | | | | | | | | | | | | | | |
| **surprise** | 0,66 | 0,67 | 0,05 | 0,38 | 1,00 | | | | | | | | | | | | | | | | | | | |
| **jaccard** | 0,94 | 0,36 | 0,26 | 0,41 | 0,55 | 1,00 | | | | | | | | | | | | | | | | | | |
| **coefficient** | 0,28 | 0,56 | 0,58 | 0,80 | 0,39 | 0,49 | 1,00 | | | | | | | | | | | | | | | | | |
| **cosine** | 0,91 | 0,55 | 0,28 | 0,54 | 0,61 | 0,97 | 0,58 | 1,00 | | | | | | | | | | | | | | | | |
| **jmeasure** | 0,38 | 0,52 | 0,38 | 0,76 | 0,44 | 0,56 | 0,91 | 0,62 | 1,00 | | | | | | | | | | | | | | | |
| **piatetsky** | 0,36 | 0,58 | 0,30 | 0,71 | 0,48 | 0,51 | 0,91 | 0,57 | 0,97 | 1,00 | | | | | | | | | | | | | | |
| **laplace** | 0,45 | 0,99 | 0,12 | 0,59 | 0,73 | 0,40 | 0,57 | 0,57 | 0,55 | 0,60 | 1,00 | | | | | | | | | | | | | |
| **specificite** | -0,17 | -0,39 | 0,44 | 0,33 | -0,34 | 0,08 | 0,47 | 0,03 | 0,33 | 0,28 | -0,40 | 1,00 | | | | | | | | | | | | |
| **sebag** | 0,46 | 0,78 | 0,08 | 0,74 | 0,72 | 0,41 | 0,49 | 0,55 | 0,53 | 0,53 | 0,80 | -0,29 | 1,00 | | | | | | | | | | | |
| **fiabilite** | 0,19 | 0,55 | 0,52 | 0,83 | 0,27 | 0,38 | 0,91 | 0,50 | 0,72 | 0,72 | 0,52 | 0,55 | 0,43 | 1,00 | | | | | | | | | | |
| **satisfaction** | 0,25 | 0,48 | 0,36 | 0,78 | 0,29 | 0,38 | 0,82 | 0,49 | 0,63 | 0,66 | 0,47 | 0,52 | 0,45 | 0,92 | 1,00 | | | | | | | | | |
| **pearl** | 0,56 | 0,09 | 0,04 | 0,34 | 0,18 | 0,62 | 0,13 | 0,57 | 0,35 | 0,17 | 0,10 | 0,02 | 0,30 | 0,08 | 0,06 | 1,00 | | | | | | | | |
| **rappel** | 0,85 | 0,18 | 0,31 | 0,31 | 0,36 | 0,94 | 0,44 | 0,89 | 0,49 | 0,42 | 0,20 | 0,20 | 0,26 | 0,32 | 0,32 | 0,57 | 1,00 | | | | | | | |
| **collective** | 0,38 | 0,56 | 0,27 | 0,73 | 0,49 | 0,50 | 0,91 | 0,57 | 0,94 | 0,98 | 0,59 | 0,31 | 0,56 | 0,74 | 0,74 | 0,14 | 0,41 | 1,00 | | | | | | |
| **klosgen** | 0,33 | 0,61 | 0,40 | 0,90 | 0,40 | 0,50 | 0,95 | 0,60 | 0,92 | 0,91 | 0,61 | 0,42 | 0,59 | 0,90 | 0,83 | 0,28 | 0,41 | 0,91 | 1,00 | | | | | |
| **Loevinger** | 0,25 | 0,48 | 0,36 | 0,78 | 0,29 | 0,38 | 0,82 | 0,49 | 0,63 | 0,66 | 0,47 | 0,52 | 0,45 | 0,92 | 0,99 | 0,06 | 0,32 | 0,74 | 0,83 | 1,00 | | | | |
| **kappa** | 0,36 | 0,58 | 0,30 | 0,71 | 0,48 | 0,51 | 0,91 | 0,57 | 0,97 | 0,99 | 0,60 | 0,28 | 0,53 | 0,72 | 0,66 | 0,17 | 0,42 | 0,98 | 0,91 | 0,66 | 1,00 | | | |
| **odds** | -0,08 | -0,13 | 0,98 | 0,22 | -0,07 | 0,20 | 0,45 | 0,16 | 0,27 | 0,18 | -0,12 | 0,54 | -0,09 | 0,38 | 0,25 | 0,02 | 0,27 | 0,16 | 0,26 | 0,25 | 0,18 | 1,00 | | |
| **Qyules** | -0,05 | -0,24 | 0,49 | 0,44 | -0,13 | 0,20 | 0,59 | 0,15 | 0,44 | 0,40 | -0,23 | 0,97 | -0,16 | 0,66 | 0,62 | 0,04 | 0,25 | 0,43 | 0,53 | 0,62 | 0,40 | 0,57 | 1,00 | |
| **Yyules** | -0,05 | -0,23 | 0,56 | 0,43 | -0,13 | 0,21 | 0,60 | 0,16 | 0,44 | 0,39 | -0,22 | 0,96 | -0,15 | 0,66 | 0,62 | 0,04 | 0,26 | 0,42 | 0,52 | 0,62 | 0,39 | 0,64 | 0,99 | 1,00 |

FIG. 3.4 – Matrice des corrélations linéaires.

| | Sup | Cof | Int | Conv | Sur | Jac | Cos | Pia |
|---|---|---|---|---|---|---|---|---|
| Moy | 0,157 | 0,491 | 1,152 | 1,313 | 0,158 | 0,235 | 0,377 | 0,008 |
| Asym (g1) | 1,459 | 0,084 | 3,166 | 3,692 | 4,484 | 1,202 | 0,611 | 0,197 |
| Apla (g2) | 1,564 | -1,068 | 21,413 | 20,174 | 33,638 | 0,830 | -0,481 | 1,219 |
| | Lap | Jme | Coe | Col | Spe | Seb | Fia | Sat |
| Moy | 0,494 | 0,017 | 0,04 | 46,6 | 0,554 | 2,028 | 0,027 | 0,50 |
| Asym (g1) | 0,104 | 1,058 | 0,022 | 0,007 | -0,879 | 1,639 | 0,130 | -2,931 |
| Apla (g2) | -0,879 | 1,639 | 0,130 | 0,979 | -0,711 | 24,958 | 0,210 | 29,04 |
| | Pea | Rap | Loe | Kap | Odd | YuQ | YuY | Klo |
| Moy | 0,039 | 0,335 | 0,012 | 0,016 | 1,610 | 0,079 | 0,043 | 0,02 |
| Asym (g1) | 1,463 | 0,768 | -2,522 | 0,197 | 3,425 | -0,104 | -0,005 | ,778 |
| Apla (g2) | 1,991 | -0,492 | 18,710 | 1,219 | 19,933 | -0,527 | -0,065 | ,648 |

TAB. 3.5 – Distribution : Estimation des coefficients d'asymétrie et d'aplatissement.

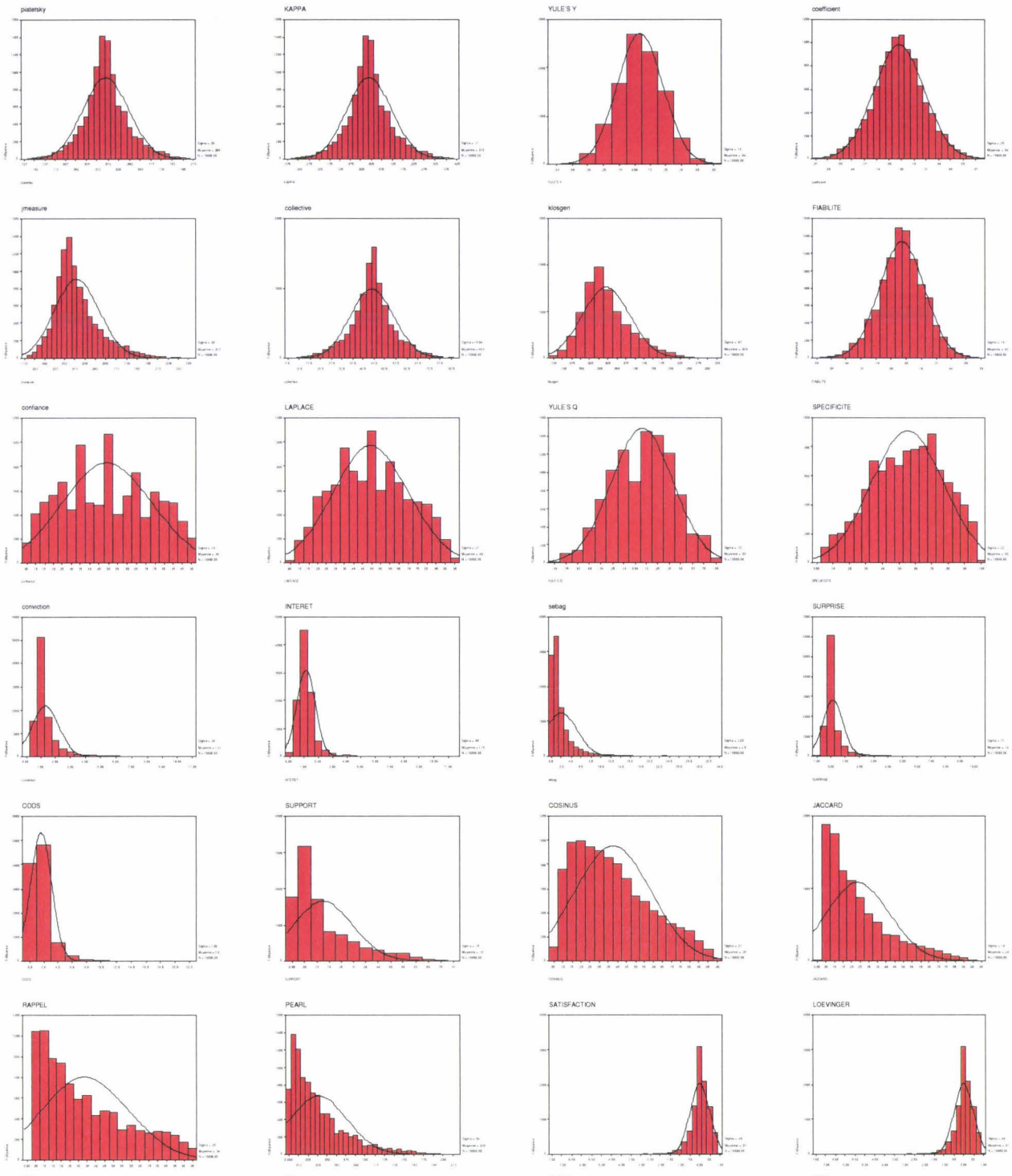
Cette matrice donne une première indication sur la nature des relations linéaires entre ces différents critères par rapport à l'ensemble des règles. On peut par exemple ainsi remarquer de très fortes corrélations entre Cosinus et Jaccard (0,97), entre J-Measure et Piatestky (0,97), entre *confiance* et *laplace* (0,99) et entre Kappa et Piatestky (0,99).

Le *support* peut être rattaché au premier groupe via sa corrélation avec Jaccard. De même Phi-Coefficient et Conviction peuvent être associés au deuxième groupe. Les critères Intérêt et Surprise sont dans l'immédiat assez atypiques.
Une première analyse en composantes principales a été effectuée sur cette matrice de corrélations.

Les figures ci-après 3.5, 3.6, 3.7 et 3.8 représentent les cercles des corrélations suivant les deux principaux facteurs. La valeur propre du facteur 1 (75,44 % d'inertie) confirme les tendances précitées et met en relief l'importance du premier axe factoriel.

Le tableau 3.7 récapitule les différentes associations dégagées lors de cette étude. Cette étude a été confirmée sur plusieurs bases de données.

C'est à partir de ces résultats que nous avons proposé une modélisation multi-objectif pour les règles d'association en utilisant plusieurs critères complémentaires permettant de mesurer la qualité des règles suivant différents aspects.




TAB. 3.6 – Histogrammes des critères avec courbe gaussienne classés par similitude.

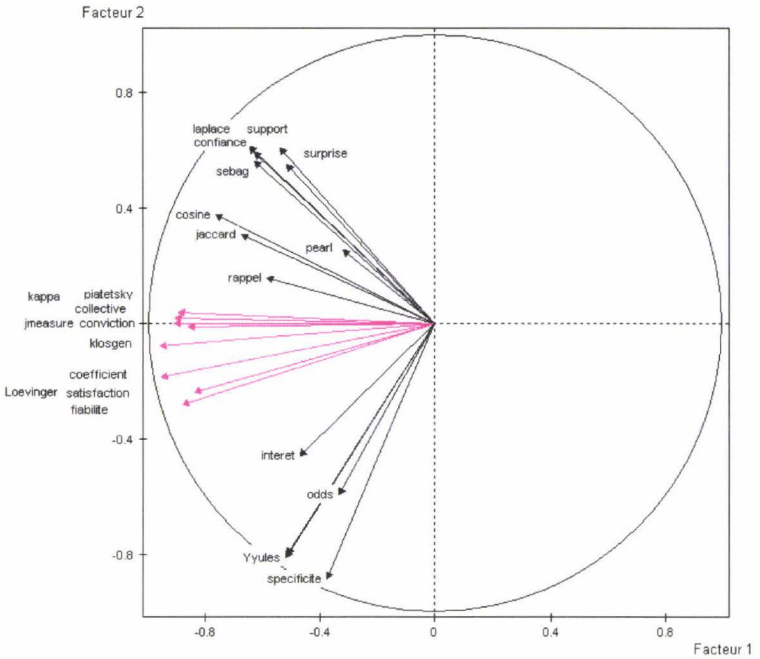



FIG. 3.5 – Cercle de corrélations : Facteur1/Facteur2.

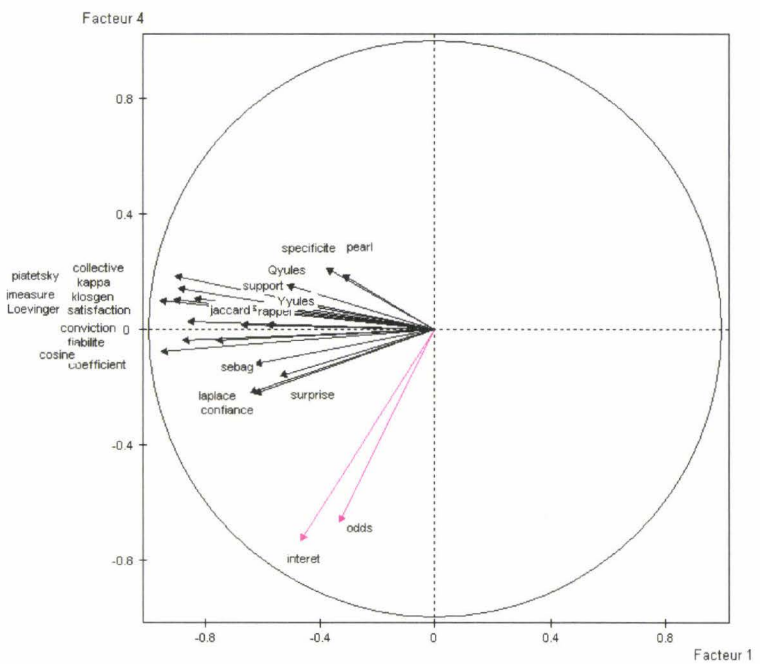



FIG. 3.6 – Cercle de corrélations : Facteur1/Facteur4.

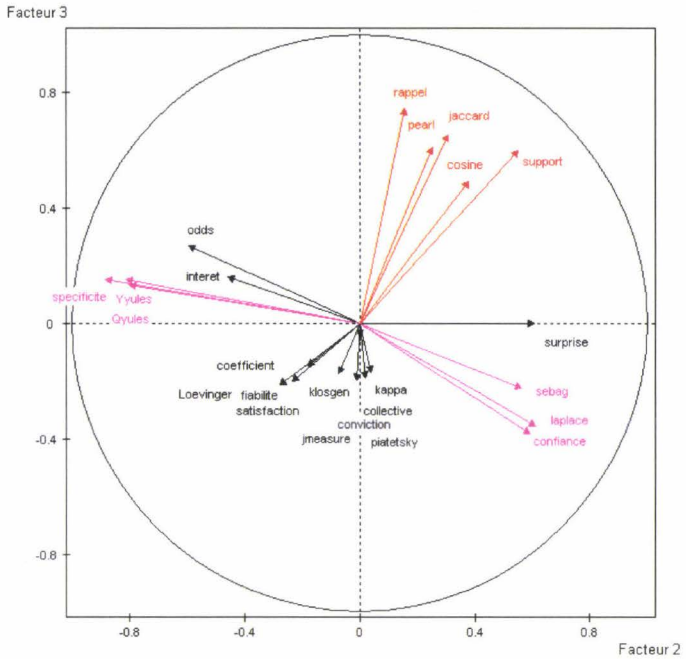



FIG. 3.7 – Cercle de corrélations : Facteur2/Facteur3.

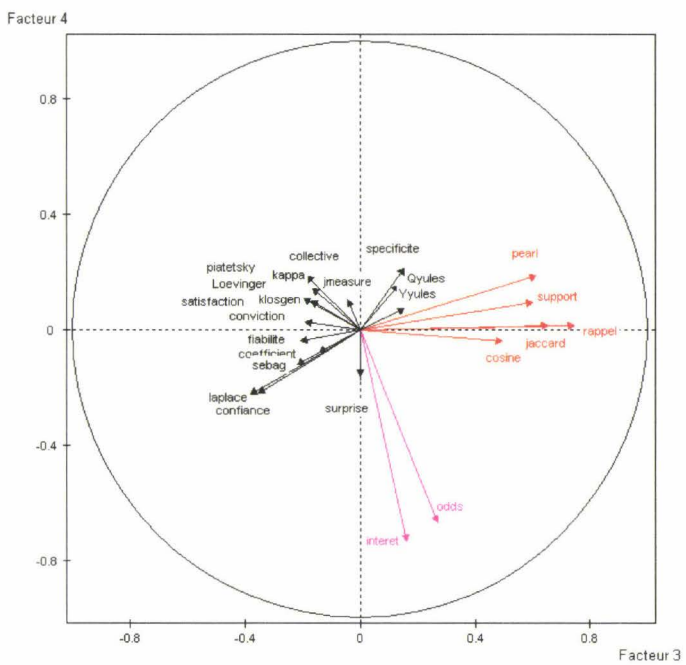



FIG. 3.8 – Cercle de corrélations : Facteur3/Facteur4.

| | | |
|---|---|---|
| Groupe 1 | **Support**, Jaccard, Cosinus, Rappel, Pearl | voir la figure 3.7 et 3.8 |
| Groupe 2 | **Confiance**, Laplace, Sebag | voir la figure 3.7 |
| Groupe 3 | **J-measure**, Conviction, Phi-Coefficient, Klosgen, Kappa, Satisfaction, Loevinger, Piatetsky, Collective, Fiabilité | voir la figure 3.5 |
| Groupe 4 | **Intérêt**, Odds | voir la figure 3.8 et 3.6 |
| Groupe 5 | **Surprise** | |
| Groupe 6 | Spécificité, Qyule's, Yyule's | voir la figure 3.7 |

TAB. 3.7 – Groupements des critères.

## 3.7 Conclusion

Dans ce chapitre, nous avons présenté en premier lieu le data mining et ses différentes techniques, ainsi que le problème la recherche de règles d'association et les méthodes de résolution de la littérature. Ensuite, nous avons exposé les différents indicateurs mesurant la qualité des règles d'association. Puis, nous avons évoqué les propriétés et préférences proposées par différents auteurs : Tan et al. [TKS02], Hilderman [HH99], V. Shi et al. [SDP01] et Piatetsky-Shapiro [PS91]. Ces travaux apportant un éclairage probabiliste et théorique pour caractériser une bonne mesure de qualité d'une règle d'association.

Finalement, nous avons proposé une approche statistique. Le but de cette étude est de mettre en évidence les corrélations existantes entre les mesures afin de regrouper les critères ayant le même comportement (et mesurent donc les mêmes propriétés) et déterminer un ensemble cohérent de critères complémentaires (nous avons trouvé 6 groupes de critères).

Nous proposons donc, dans les chapitres suivants, d'aborder le problème de recherche de règles d'association comme un problème d'optimisation combinatoire multi-objectif, pour lequel un algorithme évolutionnaire sera exposé.

Pour le reste de cette thèse, nous nous plaçons dans le cadre de problèmes d'optimisation où il n'existe pas de modèle de préférences sur les critères en utilisant cinq objectifs (un représentant par groupe pour les cinq premiers groupes). Ces critères choisis sont : le *support*, la *confiance*, la *J-mesure*, l'*intérêt* et la *surprise*.

# Chapitre 4

# L'optimisation combinatoire multi-objectif

Sommaire


4.1 Introduction . . . . . 61
4.2 Définitions et vocabulaire . . . . . 63
4.3 Algorithmes de résolution . . . . . 64
4.4 Classification des approches . . . . . 72
4.5 Evaluation de performances en optimisation multi-objectif . . 79
4.6 Conclusion . . . . . 86


## 4.1 Introduction

Dans le chapitre précédent, nous avons réalisé une analyse statistique sur les critères de qualité des règles d'association. Ainsi, nous avons regroupé les critères ayant le même comportement ce qui nous a permis de déterminer un ensemble cohérent de cinq critères complémentaires.

On peut donc formuler le problème de recherche de règles d'association sous la forme d'un problème d'optimisation combinatoire multi-objectif, où l'on cherche à optimiser simultanément plusieurs critères complémentaires.

Dans ce chapitre, nous présentons l'optimisation combinatoire multi-objectif. Nous commençons par définir quelques notions de l'optimisation multi-objectif, puis nous présentons différentes méthodes de résolution existantes dans ce domaine. Nous introduisons les méthodes à base de recherche locale et les approches évolutionnaires. Nous décrivons aussi les principales mesures des performances des algorithmes multi-objectifs.

Nous terminons par présenter un outil graphique Guimoo (a Graphical User Interface for Multi Objective Optimization) présentant une interface graphique pour les problèmes d'optimisation multi-objectifs. Il offre une visualisation 2D et 3D des fronts de problèmes

d'optimisation multi-objectif. et permet aussi l'évaluation de performances des algorithmes d'optimisation multi-objectif.

### 4.1.1 Optimisation combinatoire

L'optimisation combinatoire occupe une place très importante en recherche opérationnelle, en mathématiques discrètes et en informatique. Son importance se justifie par la difficulté des problèmes d'optimisation [PS82] et par le nombre d'applications formulées sous la forme d'un problème d'optimisation combinatoire.

Un problème d'optimisation combinatoire est défini par un ensemble d'instances. A chaque instance du problème est associé un ensemble discret de solutions $S$, un sous-ensemble $X$ de $S$ représentant les solutions admissibles (réalisables) et une fonction de coût $f$ (ou fonction objectif) qui affecte à chaque solution $s \in X$ la valeur (nombre réel ou autre type) $f(s)$. Résoudre un tel problème (plus précisément une telle instance du problème) consiste à trouver une solution $s \in X$ optimisant la valeur de la fonction de coût $f$. Une telle solution $s$ s'appelle une solution optimale ou un optimum global. Un problème d'optimisation combinatoire peut donc être défini comme suit :

**Définition 1** *Une instance $I$ d'un problème de minimisation est un couple $(X,f)$ où $X \subseteq S$ est un ensemble fini de solutions admissibles, et $f$ une fonction de coût (ou objectif) à minimiser $f : X \rightarrow \Re$. Le problème est de trouver $s^* \in X$ tel que $f(s^*) \leq f(s)$ pour tout élément $s \in X$.*

Notons que d'une manière similaire, on peut également définir les problèmes de maximisation en remplaçant simplement $\leq$ par $\geq$.

L'optimisation combinatoire trouve des applications dans des domaines aussi variés que la gestion, l'ingénierie, la conception, la production, les télécommunications, les transports, l'énergie, les sciences sociales et l'informatique elle-même.

### 4.1.2 Problème d'optimisation multi-objectif

L'optimisation multi-objectif consiste à optimiser plusieurs composantes d'un vecteur de fonctions de coût, chaque composante de ce vecteur correspondant à un objectif. Un Problème d'Optimisation Multi-objectif (*Multiobjective Optimization Problem - MOP*), peut être défini comme suit[1] :

$$(MOP) = \begin{cases} min\ F(x) = (f_1(x), f_2(x), \ldots, f_n(x)) \\ t.q.\ x \in D \end{cases} \tag{4.1}$$

$n \geq 2$ est le nombre de fonctions objectifs, $x = (x_1, x_2, \ldots, x_r)$ le vecteur de variables de décision, $D$ l'espace des solutions réalisables (espace décisionnel), et $F(x)$ le vecteur

[1] On parle ici de minimisation des objectifs, le parallèle avec les problèmes de maximisation se fait aisément.

des $n$ fonctions objectifs à optimiser. L'ensemble $O = F(D)$ correspond à l'ensemble des points atteignables dans l'espace objectif, et $f_x = (f_1(x), f_2(x), \ldots, f_n(x))$ est un point de l'espace objectif (Fig.4.1).

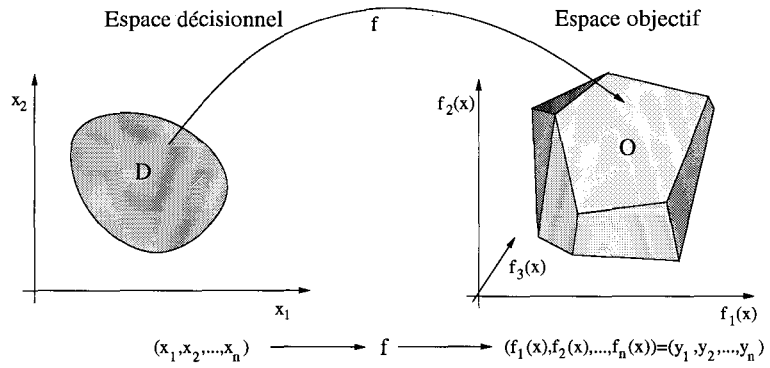



FIG. 4.1 – Problème d'optimisation multi-objectif : exemple avec 2 variables de décision et 3 fonctions objectifs.

## 4.2 Définitions et vocabulaire

### 4.2.1 Relations d'ordre et de dominance

Afin de comparer les solutions entre elles, il est nécessaire de définir une relation d'ordre entre les éléments (une solution peut être meilleure qu'une autre sur certains objectifs et moins bonne sur les autres). Dans le cas des problèmes d'optimisation multiobjectif, ces relations d'ordre sont appelées relations de dominance.

**Définition 2** *Une solution $x_i$ domine une solution $x_j$ si et seulement si :*
$\forall k \in [1..n], f_k(x_i) \leq f_k(x_j)$ *et* $\exists k \in [1..n]$ *tq* $f_k(x_i) < f_k(x_j)$,
*où $n$ est le nombre d'objectifs à optimiser.*

Le fait qu'une solution $x_i$ domine une solution $x_j$ sera noté $x_i \preceq x_j$. Si $x_i$ est meilleure que $x_j$ pour tous les objectifs, on notera alors $x_i \prec x_j$. Lorsque l'on a ni $x_i \preceq x_j$, ni $x_j \preceq x_i$, on notera alors $x_i \sim x_j$.

La notion d'optimalité la plus généralement admise est celle introduite par Edgeworth en 1881 [Edg81], généralisée plus tard par Pareto en 1896 [Par96]. Le terme le plus employé pour s'y référer est celui d'*optimum de Pareto.*

### 4.2.2 Front Pareto

**Définition 3** *Une solution est dite Pareto optimale si elle n'est dominée par aucune autre solution admissible.*

Les solutions Pareto représentent les solutions de meilleurs compromis entre les objectifs. Elles forment donc l'ensemble des solutions optimales du problème. Ainsi, il convient de rechercher des populations de solutions dites Pareto optimales. L'ensemble des solutions Pareto optimales est aussi appelé frontière Pareto (Fig.4.2).

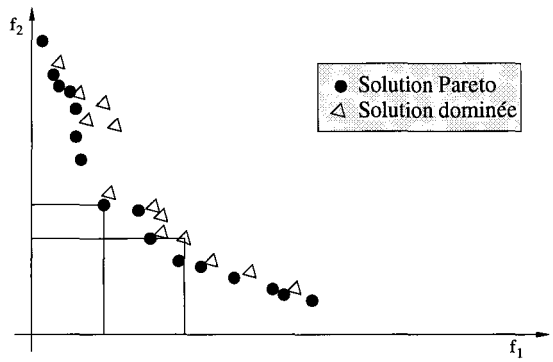



FIG. 4.2 – Solutions Pareto optimales (cas pour la minimisation de 2 objectifs).

**Définition 4** *Pour un problème d'optimisation multi-objectif donné $F(x)$, l'ensemble Pareto optimal $\mathcal{PO}^*$ est défini comme suit :*

$$\mathcal{PO}^* = \{x \in D \mid \nexists x' \in D, F(x') \preceq F(x)\} \tag{4.2}$$

L'image de l'ensemble Pareto optimal dans $D$ est appelée frontière Pareto, ou surface de compromis. L'allure de cette frontière prend des formes différentes selon que les objectifs doivent être minimisés ou maximisés. Un exemple avec deux critères est représenté sur la figure 4.3.

## 4.3 Algorithmes de résolution

De nombreuses méthodes ont été développées en recherche opérationnelle et en intelligence artificielle pour résoudre ces problèmes. Ces méthodes peuvent être classées en deux grandes classes : les méthodes exactes et les méthodes approchées (heuristiques).

Le contexte de la thèse étant la résolution d'un problème d'optimisation combinatoire multi-objectif à l'aide de métaheuristiques, nous nous focalisons dans la section suivante sur deux grandes classes de métaheuristiques, à savoir les méthodes de recherche locale et les algorithmes évolutionnaires à base de population.

Le principe essentiel d'une méthode exacte consiste généralement à énumérer l'ensemble des solutions. Pour améliorer l'énumération des solutions, une telle méthode dispose de techniques pour détecter le plus tôt possible les bornes et d'heuristiques spécifiques pour orienter les différents choix. Parmi les méthodes exactes, on trouve les techniques de

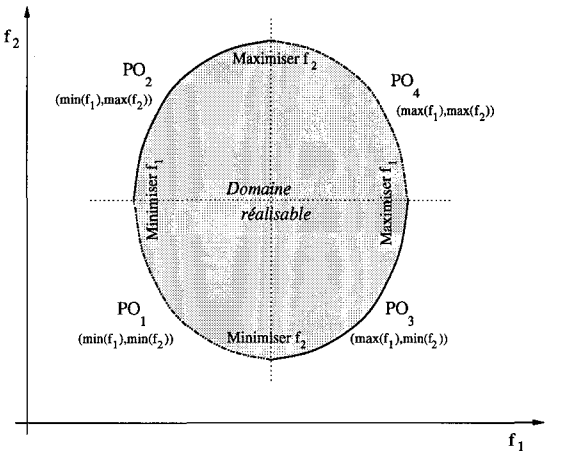



FIG. 4.3 – Allure de la frontière Pareto selon que l'on considère une maximisation ou une minimisation des différents objectifs (cas avec deux objectifs).

séparation et évaluation ou les algorithmes avec retour arrière. Les méthodes exactes rencontrent généralement des difficultés face aux applications de taille importante car le temps de calcul nécessaire pour trouver une solution risque d'augmenter exponentiellement avec la taille du problème.

Les méthodes approchées constituent une alternative très intéressante pour traiter les problèmes d'optimisation de grande taille si l'optimalité n'est pas primordiale. En particulier, les métaheuristiques sont adaptables et applicables à une large classe de problèmes. Les métaheuristiques sont représentées essentiellement par les méthodes de recherche locale comme la descente locale, le recuit simulé, GRASP et la recherche tabou, et les algorithmes évolutionnaires comme les algorithmes génétiques, la recherche dispersée et les colonies de fourmis.

### 4.3.1 Méthodes de recherche locale

Les méthodes de recherche locale sont fondées sur la notion de voisinage. Elles commencent par une solution initiale de l'espace de recherche et réalisent des transitions dont le but est d'atteindre un optimum local.

Nous allons donc introduire d'abord la notion de voisinage ainsi que quelques méthodes basées sur cette notion.

**Définition 5** *Soit $X$ l'ensemble des configurations admissibles d'un problème, on appelle voisinage toute application $N : X \rightarrow X$ définissant, pour chaque configuration $s \in X$, un ensemble $N(s) \subseteq X$ de configurations "proches" de $s$. On appelle mécanisme d'exploration du voisinage toute procédure qui précise comment la recherche passe d'une configuration $s \in X$ à une configuration $s\prime \in N(s)$. Une configuration $s$ est un optimum (minimum) local par rapport au voisinage $N$ si $f(s) \leq f(s\prime)$ pour toute configuration $s\prime \in N(s)$.*

Les méthodes de voisinage diffèrent essentiellement entre elles par le type de voisinage utilisé et la stratégie de parcours de ce voisinage. La *descente locale* est un exemple simple de cette classe de méthodes. Le recuit simulé est une méthode inspirée de la thermodynamique, ou plus exactement de la physique statistique. Pour ce qui est de la recherche tabou, elle commence par une solution initiale et tente de l'améliorer au fur à mesure que la recherche progresse, elle opère de la même façon que les méthodes descendantes, sauf qu'elle a la capacité d'échapper aux optima-locaux. GRASP est une méthode constructive permettant de construire une solution de proche en proche, en partant d'une solution partielle qui est améliorée. Elle est basée sur une fonction gloutonne (greedy) et la descente locale.

#### 4.3.1.1 Descente locale

La descente locale est une méthode d'amélioration itérative simple permettant d'atteindre un optimum local. La descente pour un problème de minimisation peut être définie simplement par l'algorithme suivant :

**Algorithme 4** Méthode de descente locale

**Entrées:** : $x_0$,$f$,$N$
  $x \leftarrow x_0$
  **répéter**
    choisir la meilleure solution $x\prime$ du voisinage $N(x)$
  **jusqu'à** $f(x\prime) > f(x)$
  Retourner $x$.

Où $N$ est la fonction de voisinage, $f$ la fonction d'évaluation, et $x_0$ la solution initiale servant de point de départ à l'algorithme. Ainsi le plus proche optimum local de $x_0$ est trouvé. Mais celui-ci peut être loin de l'optimum global, et être une mauvaise approximation de cet optimum. Pour essayer de se rapprocher de l'optimum global, plusieurs techniques sont envisageables. La technique couramment utilisée est celle du **multi-départs**. Elle consiste à ré-exécuter l'algorithme de descente en prenant un autre point de départ. Comme l'exécution de ces méthodes est souvent très rapide, on peut alors inclure cette répétition au sein d'une boucle. On obtient alors un algorithme de type **multi-départs-descente** décrit par l'algorithme 5 :

La descente permet, dans un temps de développement assez court, de calculer rapidement des premières approximations de l'optimum global.

#### 4.3.1.2 Recuit simulé

L'origine du recuit simulé, remonte à 1953 quand il été utilisé pour simuler sur un ordinateur le processus du recuit simulé de cristaux. L'idée pour appliquer cette méthode aux problèmes d'optimisation est apparue bien après [KGV83].

Le recuit simulé est dérivé des méthodes de physique statistique. Il est basé sur une analogie faite avec le procédé du recuit physique utilisé en métallurgie qui consiste à

**Algorithme 5** Pseudo-code de l'algorithme multi-départs descente

**Entrées:** : $N$,$f$
  $f(min) \leftarrow +\infty$
  **répéter**
    $x_0 \leftarrow$ solution aléatoire
    $x = DESCENTE(x_0, f, N)$
    **si** $f(x) < f(min)$ **alors**
      $min \leftarrow x$
    **finsi**
  **jusqu'à** critère d'arrêt satisfait
  Retourner $min$

chauffer un matériau à haute température et à le faire refroidir très lentement afin de laisser le système atteindre son énergie minimale, car il est bien connu qu'un refroidissement rapide ou brusque entraîne un blocage du système.

Soit un système d'atomes en mouvement avec une énergie $E$, et une température $t$. Un atome est choisi aléatoirement, on lui applique un déplacement aléatoire, et $\Delta E$ est la variation en énergie obtenue du système. Si $\Delta E$ est négatif, le déplacement est accepté et $E$ se réduit en $E+\Delta E$. Par contre si $\Delta E$ est positif, on calcule la probabilité donnée par la règle de Boltzman : $\rho = exp(\frac{\Delta E}{\kappa t})$, où $\kappa$ est la constante de Boltzman, $t$ est la température du système évaluée à l'échelle de Kelvin. $\Delta E$ est comparé à une valeur $x$ aléatoire uniformément distribuée dans [0,1]. Pour que la valeur de $x$ soit acceptée, elle doit être inférieure à $\rho$.

Afin d'utiliser l'algorithme du recuit simulé pour un problème d'optimisation combinatoire particulier, il y a un nombre d'éléments à définir :

- Le voisinage de toute solution doit être défini ainsi que le choix de détermination de la fonction objectif à minimiser selon le problème. La solution initiale doit aussi être générée.
- Les paramètres : La valeur initiale de la température $T$. Une fonction de température $\alpha(t)$ pour déterminer comment s'effectue le changement de température.
  Nous donnons maintenant un pseudo-code d'un algorithme type du recuit simulé pour un problème de minimisation (algorithme 6) :

où $T_0$ est la température initiale, *Seuil* le seuil minimal que la température peut atteindre, $\alpha$ la fonction diminuant la température à certains paliers, $it_{palier}$ le nombre d'itérations à effectuer dans un palier, $N$ la fonction de voisinage, $f$ la fonction d'évaluation, et $x_0$ la solution initiale servant de point de départ à l'algorithme. Le nombre d'itérations ($it_{palier}$) devant être atteint pour effectuer un changement de palier est fixe.

Le recuit simulé constitue, parmi les méthodes de voisinage, l'une des plus anciennes. Il a acquis son succès essentiellement grâce à des résultats pratiques obtenus sur de nombreux problèmes NP-difficiles. Des exemples de ces applications sont présentés dans [AK89, KAJ94, Col88].

**Algorithme 6** Pseudo-code du recuit simulé

**Entrées:** $T_0$, Seuil, $\alpha$, $it_{palier}$, $N$, $f$, $x_0$

```
x ← x0
T ← T0
tantque T > Seuil faire
  nombre_iterations ← 0
  tantque nombre_iterations < it_palier faire
    nombre_iterations ← nombre_iterations + 1
    choisir x' ∈ N(x)
    ΔE ← f(x') − f(x)
    si ΔE < 0 alors
      x ← x'
    sinon
      Tirer de manière aléatoire une probabilité Pr
      si Pr < exp(ΔE/T) alors
        x ← x'
      finsi
    finsi
  fin tantque
  T ← α(T) //Diminuer la température de T
fin tantque
Retourner x
```

#### 4.3.1.3 Algorithme glouton aléatoire (GRASP)

La métaheuristique GRASP (Greedy Randomized Adaptive Search Procedure) est une méthode multidéparts pour la solution approchée de problèmes difficiles d'optimisation combinatoire [FR94, PR02]. GRASP combine une heuristique gloutonne (phase de construction) et une recherche aléatoire (descente locale). La meilleure solution est gardée comme résultat.

Dans la phase de construction, une solution est construite itérativement (chaque itération ajoute un élément dans la solution partielle courante). Le choix de l'élément suivant à ajouter est déterminé en ordonnant tous les éléments dans une liste de candidats en respectant une certaine fonction gloutonne. Cette fonction mesure l'avantage de sélectionner chaque élément dont la valeur est mise à jour à chaque itération. L'élément sélectionné de la liste n'est pas nécessairement le premier de la liste mais l'un des meilleurs. On obtient ainsi une solution différente à chaque itération de GRASP. Cette étape de construction continue jusqu'à ce qu'une solution complète soit obtenue.

La solution donnée par la première phase n'est pas garantie comme optimum local, c'est pour cela qu'on fait appel à une recherche locale à partir de la solution obtenue pour essayer de l'améliorer.

Cette méthode a été appliquée avec succès à plusieurs problèmes d'optimisation [Kon94, kJ95, LFE94, FR94, RBR03, RB04].

#### 4.3.1.4 Recherche Tabou

La recherche Tabou hérite du fonctionnement de base de la descente locale mais avec beaucoup d'améliorations. Elle a été développée pour pouvoir sortir des optima locaux. Le principe de base est de poursuivre la recherche de solutions même lorsqu'un optimum local est rencontré et de permettre des déplacements qui n'améliorent pas la solution courante en utilisant le principe de mémoire pour éviter les retours en arrière (mouvements cycliques) [GL98].

La première étape consiste à se déplacer d'une solution $x$ à une autre solution $x\prime$ choisie parmi les solutions possibles de l'ensemble de voisinage $N(x)$. En atteignant un optimum local et en se déplaçant à travers l'espace des solutions pour une meilleure évolution, on peut retrouver un même optimum local, d'où l'apparition de cycles.

Une solution à ce problème est d'utiliser une structure de données appelée liste tabou comprenant toutes les solutions interdites.

Différentes stratégies de la gestion de la liste tabou ont été proposées :

Stratégie tabou : cette stratégie est une solution au problème de cycle (boucle). La liste tabou contiendra les derniers états visités auparavant. A chaque fois que l'on veut passer d'un état vers un autre, on vérifie si ce dernier n'est pas dans la liste tabou.

Stratégie de suppression de la liste : si une solution $x$ est dans la liste tabou, combien de temps devrait-elle y rester ? Ce paramètre est très important à déterminer pour que la liste tabou reste contrôlable.

La mémorisation de solutions entières serait trop coûteuse en temps de calcul et en place mémoire. Il est préférable de mémoriser des caractéristiques (un attribut) des solutions au lieu de solutions entières. Il en résulte que toutes les solutions possédant cet attribut, y compris celles qui n'ont pas encore été rencontrées, deviennent elles aussi Tabou. Pour pallier à ce problème, un mécanisme particulier, appelé l'*aspiration*, est mis en place. Ce mécanisme consiste à révoquer le statut Tabou d'une solution à certains moments de la recherche. La fonction d'aspiration la plus simple consiste à enlever le statut Tabou d'une solution si celle-ci est meilleure que la meilleure solution trouvée.

L'algorithme 7 présente un pseudo code pour la recherche Tabou pour un problème de minimisation.

Il existe aussi d'autres techniques permettant d'améliorer les performances de la méthode Tabou, en particulier, l'intensification et la diversification. L'intensification permet de se focaliser sur certaines zones de l'espace de recherche en apprenant des propriétés favorables, par exemple les propriétés communes souvent rencontrées dans les meilleurs solutions visitées. La diversification a un objectif inverse de l'intensification. En effet, elle cherche à diriger la recherche vers des zones inexplorées, en modifiant par exemple la fonction d'évaluation. L'intensification et la diversification jouent donc des rôles complémentaires.

La méthode tabou suscite un intérêt croissant depuis sa découverte et de nombreux raffinements ont été introduits dans la méthode [Glo89a, Glo89b, GL93, GKL95, GL98].

**Algorithme 7** Méthode de recherche Tabou

**Entrées:** : $f$ fonction de coût et nombre d'itérations $MaxIter$.

```
Solution courante x ← x_0 ;
Meilleure solution M ← x ;
K ← 0 ;
tantque K < MaxIter faire
  K ← K + 1
  Mise à jour de liste tabou
  Génération des candidats E par opération de voisinage
  M ← best(E)
  si (f(x) < f(M)) OU M n'est pas tabou OU M vérifie l'aspiration alors
    x ← M
  sinon
    E ← E \ M
  finsi
fin tantque
return x
```

### 4.3.2 Métaheuristiques à population de solutions

Les méthodes d'optimisation à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions en combinant des solutions entre elles pour en former de nouvelles. L'intérêt de ces méthodes est d'utiliser la population comme facteur de diversité et de fournir un ensemble de bonnes solutions, ce qui est tout à fait intéressant en optimisation multiobjectif comme nous le verrons plus tard. Parmi ces algorithmes à population, on retrouve les algorithmes génétiques, les colonies de fourmis et la recherche dispersée que nous présentons rapidement.

#### 4.3.2.1 Algorithmes génétiques

Les algorithmes génétiques forment une famille très intéressante d'algorithmes d'optimisation. Ils ont été inspirés de la génétique et de la théorie de sélection naturelle citée par Charles Darwin au 19ème siècle. Leur développement a pour but de modéliser des systèmes adaptatifs naturels et de construire des systèmes artificiels dotés des mêmes propriétés.

Les algorithmes génétiques sont l'une des méthodes de la vie artificielle. Ils ont été développés pour la première fois par John Holland, ses collègues et ses étudiants de l'université de Michigan.

L'algorithme génétique fait évoluer une population d'individus (chromosomes) en utilisant simplement les mécanismes de la sélection naturelle : les plus forts individus (au sens des critères liés à la fonction d'évaluation à optimiser) auront plus de descendants que les autres.

Plus précisément, un algorithme génétique est un algorithme itératif de recherche globale dont le but est d'optimiser une fonction d'évaluation ou d'adéquation ; pour atteindre

cet objectif, l'algorithme travaille en parallèle sur une population de points candidats appelés individus ou chromosomes. Chaque individu est constitué d'un ensemble d'éléments appelés caractéristiques ou gènes pouvant prendre plusieurs valeurs appartenant à un alphabet non nécessairement numérique. Le but est donc de trouver la meilleure combinaison de ces éléments afin d'atteindre le maximum d'adéquation.

A chaque itération, appelée génération, est crée une nouvelle population d'individus. Cette nouvelle génération est constituée généralement par des individus mieux adaptés à l'environnement (cela est désigné par la fonction d'évaluation). Ces individus sont créés en utilisant des parties des meilleurs éléments de la population précédente ainsi que des parties novatrices. Au fur et à mesure des générations, les individus vont tendre vers l'optimum de la fonction d'évaluation.

Nous détaillerons plus amplement ces algorithmes qui vont être la base de notre travail, dans la section 5.1.

#### 4.3.2.2 Recherche dispersée

La recherche dispersée est une méthode d'optimisation relativement ancienne décrite par Glover [Glo77]. Cette approche évolutionnaire a pour origine les stratégies de création de règles de décision composées et de contraintes de remplacement.
Les études récentes ont démontré les avantages pratiques de cette approche pour résoudre divers problèmes d'optimisation.

La recherche dispersée opère sur une population de solutions et emploie des procédures pour combiner ces solutions afin d'en créer de nouvelles.

Comme les algorithmes génétiques, la recherche dispersée commence son processus par la création aléatoire d'un ensemble initial. A partir de cet ensemble qui s'appelle population initiale, la procédure de recherche se déclenche. Pratiquement le choix de l'ensemble respecte la cardinalité et la diversité des éléments.

Dans les algorithmes génétiques, on combine les éléments deux à deux pour générer deux autres éléments. La combinaison se fait par les opérateurs génétiques (mutation, crossover,...). Par contre, la recherche dispersée consiste à faire la combinaison suivant plusieurs types. Dans le type de base, on combine les éléments deux à deux. Dans le type suivant, la combinaison sera entre trois éléments, et ainsi de suite. L'élément construit par la combinaison doit être amélioré par une autre heuristique [Glo98].

#### 4.3.2.3 Colonies de fourmis

L'histoire de l'intelligence en essaim remonte à l'étude du comportement de fourmis à la recherche de nourriture au départ de leur nid, par Goss, Deneubourg et leur équipe [DPV83, DG89].
En se déplaçant du nid à la source de nourriture et vice-versa (ce qui, dans un premier temps, se fait essentiellement d'une façon aléatoire), les fourmis déposent au passage sur le sol une substance odorante appelée phéromone, ce qui a pour effet de créer une piste chimique. Les fourmis peuvent sentir ces phéromones qui ont un rôle de marqueur de che-

min : quand les fourmis choisissent leur chemin, elles ont tendance à choisir la piste qui porte la plus forte concentration de phéromones. Cela leur permet de retrouver le chemin vers leur nid lors du retour. D'autre part, les odeurs peuvent être utilisées par d'autres fourmis pour retrouver les sources de nourriture détectées par leurs consoeurs.

Il a été démontré expérimentalement que ce comportement permet l'émergence des chemins les plus courts entre le nid et la nourriture, à condition que les pistes de phéromones soient utilisées par une colonie entière de fourmis.
Le système de fourmis (Ants System - AS) est une méthode d'optimisation basée sur ces observations proposées par Dorigo [DMC91, DMC96, Dor92]. Le système de fourmis a été employé avec succès sur des nombreux problèmes (voyageur de commerce, affectation quadratique, ...) mais les auteurs ont remarqué que l'AS n'a pas un comportement très exploratoire ce qui a conduit les auteurs à utiliser des hybridations du système de fourmis avec des recherches locales.
Parmi les applications de tels algorithmes, nous pouvons citer : Optimisation par colonies de fourmis appliqué au découpage de l'espace aérien européen en zones de qualification [BA05], affectation quadratique [CVH02], réseaux mobiles Adhoc [DDG04] ou routage de véhicule [BHS97].

Pour les règles d'association, Parpinelli et al. proposent AntMiner pour rechercher des règles de classification et l'appliquent à des bases de données médicales [PLF02].

Nous avons présenté dans cette partie quelques méthodes de résolution issues des métaheuristiques pour l'optimisation mono-bjectif. Ces méthodes ont montré leur efficacité pour trouver des solutions approchées satisfaisantes pour un grand nombre de problèmes. Cependant, les problèmes d'optimisation rencontrés en pratique sont rarement mono-objectif. Il y a généralement plusieurs critères (multi-objectifs) contradictoires à satisfaire simultanément. C'est d'ailleurs le cas du problème de recherche de règles d'association que nous étudions dans cette thèse.

Nous allons présenter dans la section suivante l'optimisation multi-objectif.

## 4.4 Classification des approches

Dans la littérature, il y a plusieurs travaux utilisant les méthodes exactes pour la resolution des problèmes multi-objectifs. Ces méthodes sont limitées aux problèmes de petites tailles et de deux critères au maximum. Nous citons : *méthodes par décomposition* [SRD88, UT95, VTPU98, SK99, BLDT04, LDT05], *l'algorithme A** [SW91, MM96], et la *programmation dynamique* [Whi82, CMM90].

Des méthodes heuristiques sont nécessaires pour résoudre les problèmes de grandes tailles et/ou les problèmes avec un nombre de critères supérieur à deux. Elles ne garantissent pas de trouver de manière exacte l'ensemble Pareto optimal , mais une approximation de cet ensemble. Les méthodes heuristiques peuvent être divisées en deux classes : les algorithmes spécifiques à un problème donné qui utilisent des connaissances du domaine [GLCM94], et d'autre part les algorithmes généraux (*métaheuristiques*) applicables à une

grande variété de MOP. Notre intérêt porte sur les métaheuristiques.

Plusieurs adaptations de *métaheuristiques* ont été proposées dans la littérature pour la résolution de MOP et la détermination des solutions Pareto : le *recuit simulé* [Ulu93], la *recherche tabou* [GMF96] et les *algorithmes évolutionnaires* (algorithmes génétiques [SD95a, Fon95, Bas05], stratégies évolutionistes [Kur91]).

Un grand nombre d'approches existent pour résoudre les problèmes multi-objectifs. Certaines utilisent des connaissances du problème pour fixer des préférences sur les critères et ainsi contourner l'aspect multicritère du problème. D'autres mettent tous les critères au même niveau d'importance. Plusieurs états de l'art peuvent être consultés notamment dans [Tal00, UT94, CS02, EG00, DK01].

Les approches utilisées pour la résolution de MOP peuvent être classées en trois catégories (fig.4.4) :

- **Approches scalaires** (basées sur la transformation du problème en un problème mono-objectif) : Cette classe d'approches comprend par exemple les méthodes basées sur l'agrégation qui combinent les différentes fonctions coût $f_i$ du problème en une seule fonction objectif $F$. Ces approches nécessitent pour le décideur d'avoir une bonne connaissance de son problème.
- **Approches non scalaires et non-Pareto** : Ces approches ne transforment pas le MOP en un problème mono-objectif. Elles utilisent des opérateurs de recherche qui traitent séparément les différents objectifs.
- **Approches Pareto** : Les approches Pareto utilisent directement la notion d'optimalité Pareto dans leur processus de recherche. Le processus de sélection des solutions générées est basé sur la notion de non-dominance.

Dans les sections suivantes, nous présentons respectivement les trois classes de méthodes.

### 4.4.1 Méthodes scalaires

Dans la résolution de MOP, plusieurs méthodes traditionnelles transforment le MOP en un problème mono-objectif. Parmi ces méthodes on trouve les méthodes d'agrégation, les méthodes $\epsilon$-contrainte et les méthodes avec vecteur cible (but).

#### Méthode d'agrégation

C'est l'une des premières méthodes utilisée pour la génération de solutions Pareto optimales. Elle consiste à transformer le problème multi-objectif en un problème à un objectif en combinant les différentes fonctions coût du problème en une seule fonction objectif globale généralement de façon linéaire [HM79, SD95a].

Cette approche a largement été utilisée dans la littérature à l'aide de différentes métaheuristiques, nous citons par exemple les travaux : [Coe98a, LBF98, DJL95, UTFT98].

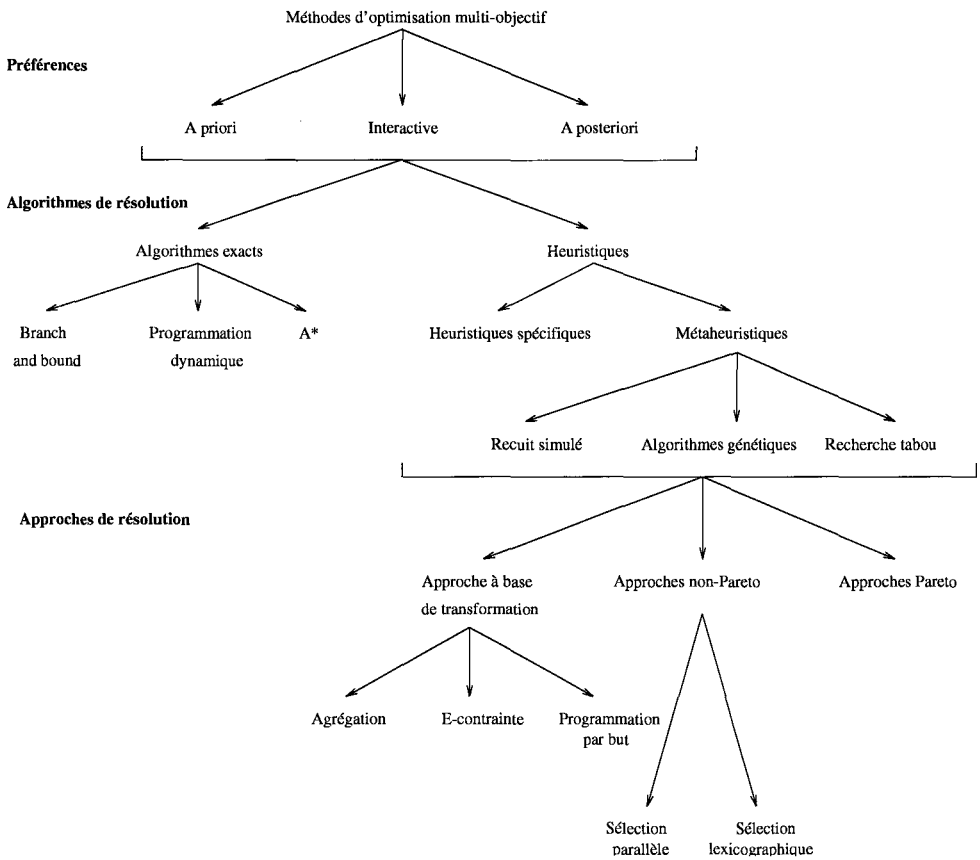



FIG. 4.4 – Classification des méthodes d'optimisation multi-objectif.

#### Méthode $\epsilon$-contrainte

Une autre façon de transformer un problème d'optimisation multi-objectif en un problème mono-objectif est de convertir $n-1$ des $n$ objectifs du problème en contraintes et d'optimiser l'objectif restant.

Des exemples d'applications de ces méthodes peuvent être trouvés dans les travaux [LE97, Sch95, VSM+97, RER94, HJRF94].

#### Méthodes avec vecteur cible

Dans ces méthodes, un ensemble de buts (ou cibles) que l'on désire atteindre est défini. La méthode d'optimisation essaie alors de minimiser la différence entre les solutions envisagées et les buts. Ces méthodes, bien que travaillant par agrégation des objectifs, permettent de générer des solutions non-supportées. Différentes approches sont envisageables pour ce type de méthode, comme celles du but à atteindre [WLK92], du but programmé [MPT00], ou du min-max [Coe98b].

### 4.4.2 Approches non-Pareto et non-scalaires

En général, les approches non-Pareto basées sur les populations de solutions possèdent un processus traitant séparément les différents objectifs.

#### Sélection parallèle

Le premier travail consistant à utiliser les AGs pour résoudre des MOP est celui de Schaffer [Sch85]. L'algorithme développé VEGA (Vector Evaluated Genetic Algorithm) sélectionne les individus de la population courante suivant chaque objectif, indépendamment des autres. A chaque génération, la population est donc divisée en un nombre de sous-populations qui est égal au nombre d'objectifs de la fonction coût. Chaque sous-population $i$ est sélectionnée suivant l'objectif $f_i$. L'algorithme VEGA sélectionne les individus selon chaque objectif de manière indépendante (sélection parallèle) et applique les opérateurs génétiques (mutation, crossover).

#### Méthode lexicographique

Cette méthode, proposée par Fourman [Fou85], classe les objectifs en fonction d'un ordre d'importance proposé par le décideur. Ensuite les fonctions objectif sont traitées dans cet ordre pour obtenir l'optimum. L'optimisation séquentielle des différents objectifs aboutit à la découverte d'une seule solution optimale. Des variantes de la méthode peuvent être définies afin d'en découvrir plusieurs, mais la méthode reste assez inappropriée pour obtenir ou approcher le front Pareto correspondant à un problème multi-objectif.

### 4.4.3 Approches Pareto

Les approches Pareto utilisent directement la notion de dominance dans la sélection des solutions générées, contrairement aux autres approches qui utilisent une fonction d'utilité ou traitent séparément les différents objectifs. Cette idée a été introduite initialement dans les AGs par Goldberg [Gol89]. Le principal avantage de ces approches est qu'elles sont capables de générer des solutions Pareto optimales dans les portions concaves de la frontière Pareto.

Les AGs ont été largement utilisés pour la résolution de MOP, étant donné qu'ils travaillent sur une population de solutions. Deux objectifs doivent être pris en compte dans la résolution d'un MOP (voir figure 4.5) :

- Converger vers la frontière Pareto : la plupart des travaux de recherche sur l'application des AGs aux MOP se sont concentrés sur l'étape de sélection en utilisant les *méthodes de ranking* afin d'établir un ordre basé sur la notion de dominance (*rank*) entre les individus (**Sélection Pareto**).
- Trouver des solutions diversifiées dans la frontière Pareto : les *méthodes de maintien de la diversité*, par la formation de niches écologiques et d'espèces, peuvent être particulièrement utiles pour stabiliser des sous-populations multiples le long de la frontière Pareto.

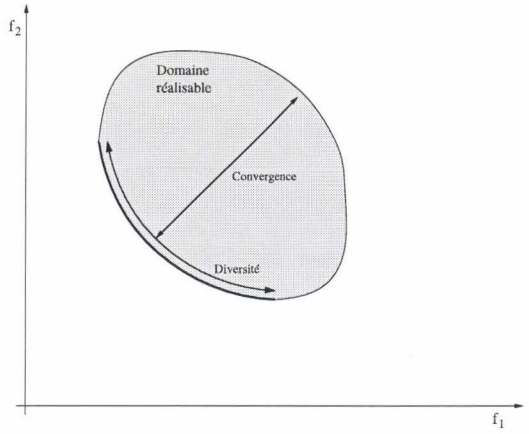



FIG. 4.5 – Les deux buts de l'optimisation multi-objectif.

Ainsi, pour appréhender un problème d'optimisation multi-objectif à l'aide des algorithmes génétiques, différents mécanismes doivent être mis en place.

#### 4.4.3.1 Méthodes de ranking

Plusieurs méthodes de ranking ont été utilisées :

- **NSGA (Non-dominated Sorting Genetic Algorithm)** : cette première procédure de ranking a été initialement proposée par Goldberg [Gol89] et implémentée

par Srinivas et Deb [SD95a]. Elle a été utilisée dans les AGs pour la résolution de plusieurs MOP [ZG99, AKC99, HWOS97, PM98b]. Actuellement, elle est utilisée pour la majorité des problèmes d'optimisation multi-objectifs (deuxième version NSGA-II).

Tous les individus non dominés de la population possèdent le rang 1. Ces individus sont ensuite enlevés de la population, et l'ensemble suivant d'individus non dominés est identifié et on leur attribue le rang 2 (fig.4.6). Ce processus est réitéré jusqu'à ce que tous les individus de la population aient un rang.

- **NDS (Non Dominated Sorting)** : dans cette méthode, le rang d'un individu est le nombre de solutions dominant l'individu plus un [FF95b]. Considérons par exemple un individu $c_i$ à la génération $t$, qui est dominé par $p_i^t$ individus dans la population courante. Son rang dans la population est donné par :

$$rang(c_i, t) = 1 + p_i^t$$

Un individu non dominé de la population possède donc le rang 1 (fig.5.4) [FF95c]. Les rangs associés à la méthode NDS sont toujours supérieurs à ceux de la méthode NSGA. Ce type de ranking induit donc une plus forte pression de sélection, et peut causer une convergence prématurée.

- **WAR (Weighted Average Ranking)** : les différents coûts de chaque individu sont évalués pour chaque objectif. Une liste de valeurs est établie pour chaque objectif. Ces listes sont alors triées suivant l'ordre décroissant des valeurs, en associant un ordre pour chaque solution et chaque objectif. La moyenne des rangs est finalement calculée pour chaque individu [BW97].

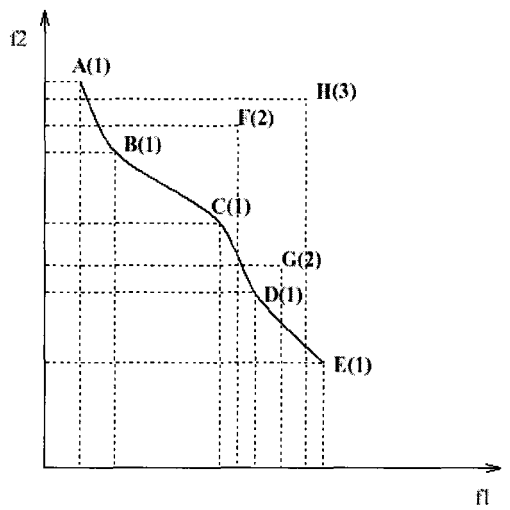



FIG. 4.6 – NSGA ranking.

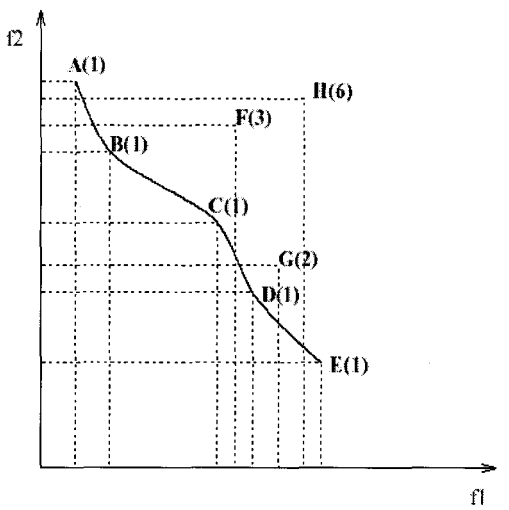



FIG. 4.7 – NDS ranking

#### 4.4.3.2 Elitisme

L'élitisme permet de conserver les meilleures solutions dans les générations futures. L'élitisme est introduit pour conserver les bonnes solutions lors du passage de la génération courante à la prochaine génération. Conserver ces solutions pour les générations futures permet d'améliorer les performances des algorithmes sur certains problèmes.

Le plus souvent l'élitisme consiste à utiliser une population externe de solutions (*ArchivePareto*) dans laquelle est stocké le meilleur ensemble des solutions non dominées découvertes pendant la recherche. Les solutions de l'*ArchivePareto* peuvent toujours être choisies par l'opérateur de sélection (sélection élitiste).

#### 4.4.3.3 Mécanisme de diversité

La diversité est une notion importante dans l'optimisation multi-objectif. Le but est de préserver la diversité de la population évoluant et d'éviter que la population converge prématurément vers une petite zone de l'espace de recherche ou de l'espace des objectifs (une partie du front Pareto).

De nombreuses techniques ont été développées. Celles-ci influent sur la probabilité de sélection, afin de privilégier certaines solutions, ou sur l'évaluation des solutions. Nous allons maintenant présenter quelques mécanismes de diversification les plus couramment intégrés aux algorithmes évolutionnaires :

**Le sharing** : Le sharing consiste à modifier la valeur de coût d'une solution, cette nouvelle valeur sera utilisée comme valeur d'adaptation par l'opérateur de sélection. En effet, le sharing permet de dégrader la fonction d'adaptation d'un individu par rapport au nombre d'individus semblables dans la population (notion de niche). Cette technique, introduite par Goldberg and Richardson [GR87a], est largement utilisée aujourd'hui.

**Le Crowding** : Holland a été le premier à suggérer l'utilisation de l'opérateur de *crowding* dans la phase de remplacement des AGs [Hol75a], pour identifier les situations dans lesquelles de plus en plus d'individus dominent les niches écologiques. Le *crowding* consiste à déterminer un représentant par niche découverte et seuls les représentants participeront aux phases de croisement, mutation et sélection.

Une première implémentation de cet opérateur a été réalisée par De Jong [Jon75]. Dans la reproduction d'un nouvel individu, l'opérateur consiste à remplacer l'individu existant le plus semblable à l'individu généré, et non pas les parents comme dans les AGs standards.

**Le parallélisme** : La plupart des travaux réalisés autour des modèles parallèles pour l'optimisation multi-objectif ont porté sur les algorithmes génétiques. Les modèles parallèles d'algorithmes génétiques peuvent être décomposés en deux classes :

- Le modèle insulaire dans lequel plusieurs sous-populations communiquent par migration d'individus (island model).
- Le modèle cellulaire qui utilise une seule population d'individus faiblement connectés, et où la sélection est locale.

Le parallélisme permet une meilleure diversité. En effet, dans le modèle insulaire, la

migration entre plusieurs sous-populations (îles) est intéressante pour maintenir la diversité en utilisant plusieurs topologies d'interconnections (anneau, tore,$\cdots$).

## 4.5 Evaluation de performances en optimisation multi-objectif

L'évaluation de performances des algorithmes d'optimisation multi-objectif n'est pas triviale, puisque ceux-ci fournissent un ensemble de solutions non dominées (front Pareto) et donc non-comparables entre elles. En particulier la comparaison d'algorithmes peut être difficile. Il est donc nécessaire d'utiliser des mesures de performance spécifiques afin d'évaluer ces algorithmes.

L'évaluation des Front Pareto est un problème délicat qui oblige à utiliser plusieurs mesures différentes, car il est impossible de représenter par une unique valeur réelle la qualité des solutions, la taille du front et la répartition des solutions sur le front. C'est pourquoi il est courant d'utiliser plusieurs mesures.

De nombreuses mesures ont été proposées. Cependant, chacune présente à la fois des avantages et des inconvénients. Dans cette section, les mesures les plus couramment utilisées sont décrites. Cependant, il est possible de trouver des études comme celle de Knowles et Corne [KC02], qui comparent plusieurs métriques.

Nous séparons les mesures (métriques) en trois classes :
- Les métriques absolues sans référence : qui indiquent la diversité et/ou la distribution des solutions du Front.
- Les métriques absolues avec une référence : ces métriques utilisent un point ou un ensemble de référence pour évaluer la qualité du front.
- Les métriques relatives : qui comparent deux fronts Pareto.

Dans la suite de cette section, $\mathcal{P}_A$ représente l'ensemble des solutions potentiellement Pareto optimales trouvé par un algorithme $A$.

### 4.5.1 Les métriques absolues sans référence

#### La métrique *Spacing*

Cette métrique, proposée par Schott [Sch95], calcule la distance relative entre deux solutions consécutives de $\mathcal{P}_A$, de la manière suivante :

$$S = \sqrt{\frac{1}{|\mathcal{P}_A|}\sum_{i=1}^{|\mathcal{P}_A|}(d_i - \bar{d})^2} \tag{4.3}$$

où

$$d_i = \min_{k \in \mathcal{P}_A \wedge k \neq i} \sum_{m=1}^{n} |f_m^i - f_m^k|$$

et $\bar{d}$ est la valeur moyenne de la distance précédente $\bar{d} = \frac{\sum_{i=1}^{|\mathcal{P}_A|} d_i}{|\mathcal{P}_A|}$. La distance $d_i$ est la valeur minimale de la somme des différences absolues des valeurs des fonctions objectifs

entre la $i^{ème}$ solution et toutes les autres solutions de l'ensemble. Il est à noter que cette distance est différente de la distance *Euclidienne* minimale entre deux solutions.

Cette métrique calcule les écarts types des différentes valeurs de $d_i$. Ainsi, si les solutions sont uniformément espacées, la distance correspondante sera faible. Donc, plus un algorithme trouve un ensemble de solutions potentiellement Pareto optimales pour lequel cette mesure est faible, meilleur est le front.

#### Métrique *Maximum spread*

Zitzler [Zit99] définit une métrique mesurant la longueur de la diagonale d'une "hyper-boîte" formée par les valeurs des fonctions objectifs extrêmes de l'ensemble potentiellement Pareto optimal généré :

$$MS = \sqrt{\frac{1}{|\mathcal{P}_A|}\sum_{m=1}^{n}\left(\frac{\max_{i=1}^{|\mathcal{P}_A|} f_m^i - \min_{i=1}^{|\mathcal{P}_A|} f_m^i}{F_m^{\max} - F_m^{\min}}\right)^2} \tag{4.4}$$

où $F_m^{\max}$ et $F_m^{\min}$ sont le maximum et le minimum pour le $m^{ème}$ objectif. Cette métrique doit être maximale. Le problème de cette mesure est qu'elle ne fournit aucune information sur la distribution exacte des solutions de compromis.

### 4.5.2 Les métriques absolues avec une référence

#### La métrique $S$

La métrique $S$ (Volume de l'espace dominé par un vecteur de référence), proposée par Zitzler [Zit99], calcule l'hypervolume de la région multidimensionnelle fermée par $\mathcal{P}_A$ et un point de référence, c'est-à-dire la taille de l'espace des objectifs que $\mathcal{P}_A$ domine.

Dans leur étude [KC02], Knowles et Corne recommandent l'utilisation de cette mesure, dont un exemple est illustré sur la figure 4.8.

#### Proportion d'erreur

Cette mesure [VL00] utilise un front de référence $PO^*$ et compte le nombre de solutions de $\mathcal{P}_A$ qui n'appartiennent pas à $PO^*$, soit :

$$ER(A) = \frac{\sum_{i=1}^{|\mathcal{P}_A|} e_i}{|\mathcal{P}_A|} \tag{4.5}$$

où $e_i = 1$ si la $i^{ème}$ solution de $\mathcal{P}_A$ appartient à $PO^*$, sinon $e_i = 0$.

Le désavantage de cette méthode est que, si aucune solution de $\mathcal{P}_A$ n'appartient à $PO^*$, elle n'apporte aucune information quant à la proximité relative de $\mathcal{P}_A$ par rapport à $PO^*$ puisque dans ce cas, quelle que soit la distance séparant $\mathcal{P}_A$ de $PO^*$, on a $ER(A) = 0$.

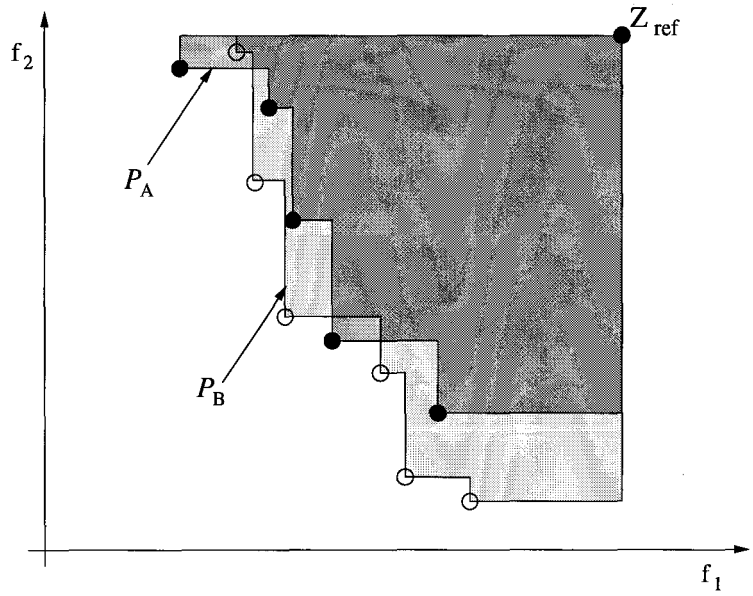



FIG. 4.8 – Métrique $S$, correspondant aux aires de dominance d'ensembles de solutions Pareto par rapport à un point de référence $Z_{ref}$.

### Distance générationnelle

Cette mesure [VL00] utilise également $PO^*$ et calcule la distance moyenne entre les solutions de $\mathcal{P}_A$ et celles de $PO^*$. Elle se calcule selon la formule suivante :

$$DG(A) = \frac{(\sum_{i=1}^{|\mathcal{P}_A|} d_i^p)^{\frac{1}{p}}}{|\mathcal{P}_A|} \tag{4.6}$$

Pour $p = 2$, le paramètre $d_i$ est la distance Euclidienne (dans l'espace des objectifs) entre la solution $i \in \mathcal{P}_A$ et le membre le plus proche de $PO^*$ :

$$d_i = \min_{k=1}^{|\mathcal{P}^*|} \sqrt{\sum_{j=1}^{n} (f_j^i - f_j^k)^2} \tag{4.7}$$

où $f_j^i$ est la valeur de la $j^{ème}$ fonction objectif de la solution $i$, et $f_j^k$ est la valeur de la $j^{ème}$ fonction objectif de la $k^{ème}$ solution de $PO^*$.

La difficulté avec cette méthode est que, s'il existe un ensemble $\mathcal{P}_A$ pour lequel il y a une fluctuation importante dans les distances, la métrique peut ne pas retourner la véritable distance. Dans un tel cas, le calcul de l'écart type de la mesure DG est nécessaire. D'autre part, si les fonctions objectifs ne sont pas de même amplitude, elles doivent être normalisées avant le calcul.

### Erreur maximale à la surface de compromis

Cette métrique permet de mesurer la distance entre $PO^*$ et l'ensemble $\mathcal{P}_A$. Elle calcule en fait la plus grande distance minimale entre les solutions de $\mathcal{P}_A$ et les solutions les plus proches de $\mathcal{P}^*$. Sa définition pour un problème à deux objectifs est la suivante :

$$EM(A) = \max_j(\min_i(|f_1^i(x) - f_1^j(x)|^p + |f_2^i(x) - f_2^j(x)|)^{\frac{1}{p}}) \tag{4.8}$$

## 4.5.3 Les métriques relatives

### La métrique $C$

Cette mesure, proposée par Zitzler [Zit99], calcule la proportion de solutions d'un ensemble potentiellement Pareto optimal $\mathcal{P}_B$ dominées par des solutions d'un ensemble potentiellement Pareto optimal $\mathcal{P}_A$ :

$$C(A,B) = \frac{|\{b \in \mathcal{P}_B | \exists a \in \mathcal{P}_A : a \prec b\}|}{|\mathcal{P}_B|} \tag{4.9}$$

$C(A,B) = 1$ signifie que toutes les solutions trouvées par l'algorithme $B$ sont dominées par celles trouvées par l'algorithme $A$. Tandis que $C(A,B) = 0$ indique qu'aucune solution engendrée par $B$ n'est dominée par une solution trouvée par $A$. Comme la relation de dominance n'est pas symétrique, $C(A,B)$ n'est pas forcément égal à $1-C(A,B)$. Il est donc nécessaire de calculer $C(A,B)$ et $C(B,A)$. Il est intéressant de noter que les cardinalités de $\mathcal{P}_A$ et $\mathcal{P}_B$ ne doivent pas forcément être identiques.

### La Contribution

Cette mesure [MTR00] se base également sur la comparaison de deux ensembles Pareto $\mathcal{P}_A$ et $\mathcal{P}_B$. Cette mesure de contribution, mise au point au sein de notre équipe, calcule la proportion de solutions non-dominées entre deux fronts.

La contribution de $A$ relativement à $B$, notée

$$Contribution(A,B) = \frac{\frac{|C|}{2} + |W_A| + |N_A|}{|C| + |W_A| + |N_A| + |W_B| + |N_B|} = \frac{\frac{|C|}{2} + |W_A| + |N_A|}{|PO|} \tag{4.10}$$

Avec $|C|$ la taille de l'ensemble des solutions potentiellement Pareto optimales communes à $A$ et $B$, $|W_A|$ le nombre de solutions de $A$ dominant celles de $B$, $|N_A|$ le nombre de solutions de $A$ qui ne dominent pas celles de $B$.

La contribution de l'algorithme $B$ relativement à l'algorithme $A$ est définie d'une manière similaire. Notons que si les deux algorithmes produisent les mêmes solutions, alors on a $Contribution(A,B) = Contribution(B,A) = 1/2$

Si toutes les solutions produites par $B$ sont dominées par les solutions produites par $A$, alors on a $Contribution(B, A) = 0$. De plus, dans le cas général, on a : $Contribution(A, B) + Contribution(B, A) = 1$.

Cette mesure permet d'avoir rapidement une idée de l'apport d'un Front par rapport à un autre Front.

### L'Entropie

Cette mesure permet d'évaluer la diversité d'une approximation $\mathcal{P}_A$ produite par un algorithme $A$ par rapport à la diversité d'une approximation $\mathcal{P}_B$ produite par un algorithme $B$ [BST02, MTR00]. On notera $PO$ l'ensemble des solutions non dominées de l'union de $\mathcal{P}_A$ et $\mathcal{P}_B$.

Dans cette mesure, une niche est associée à chaque solution. Les solutions présentes dans chaque niche sont considérées comme voisines de la solution associée à la niche. L'entropie est alors donnée par la mesure :

$$E(A/B) = \frac{-1}{\log \gamma} \sum_{i=1}^{C} \frac{1}{N_i} \frac{n_i}{C} \log \frac{n_i}{C} \tag{4.11}$$

où $N_i$ est le nombre de solutions de $\mathcal{P}_A \cup PO$ se trouvant dans la niche de la $i^{ème}$ solution de $\mathcal{P}_A \cup PO$, $C$ est le cardinal de $\mathcal{P}_A \cup PO$, $n_i$ est le nombre de solutions de l'ensemble $\mathcal{P}_A$ dans la niche de la $i^{ème}$ solution de $\mathcal{P}_A \cup PO$, et $\gamma = \sum_{i=1}^{C} \frac{1}{N_i}$ représente la somme des coefficients affectés à chaque solution.

Cette mesure permet donc une estimation de la diversité relative entre deux approximations. Toutefois, l'interprétation des résultats n'est pas toujours évidente.

### La métrique $D$

Zitzler [Zit99] propose également une métrique $D$ permettant de comparer deux fronts Pareto $\mathcal{P}_A$ et $\mathcal{P}_B$, s'appuyant sur la métrique $S$. Après avoir calculé les solutions non-dominées $PO$ à partir de l'union des deux ensembles $\mathcal{P}_A$ et $\mathcal{P}_B$, on calcule la valeur $D(\mathcal{P}_A, PO)$ qui équivaut à $S(PO - \mathcal{P}_B)$. La valeur obtenue correspond au volume dominé par $PO_1$, mais pas par $PO_2$. Il reste ensuite à comparer les valeurs $D(\mathcal{P}_A, PO)$ et $D(\mathcal{P}_B, PO)$.

$$D(A, B) = S(A + B) - S(B)$$

Lorsque $D(A, B) > 0$, la couverture de $\mathcal{P}_A$ est plus étendue que celle de $\mathcal{P}_B$.
Lorsque $D(A, B) = 0$, la couverture de $\mathcal{P}_A$ est aussi étendue que celle de $\mathcal{P}_B$.

### 4.5.4 Mesures utilisées

Pour nos expérimentations, nous ne connaîtrons pas de frontière Pareto optimale. Pour cela nous allons utiliser comme front de référence lorsque cela sera nécessaire, le front Pareto total de tous les fronts trouvés dans toutes nos expériences. Pour l'évaluation des différentes approximations Pareto calculées lors des expérimentations, nous utiliserons trois mesures : la métrique *Contribution*, la métrique $S$ et la métrique $D$.

La contribution permet d'avoir facilement une idée de la supériorité d'un algorithme par rapport à un autre. Les valeurs sont très facilement interprétables et la mesure n'est sujette à aucun paramétrage. Cependant, cette mesure ne permet pas réellement de quantifier la différence d'efficacité des deux algorithmes (elle est basée sur le nombre de solutions pareto et non sur leur qualité relative).

Afin de compléter les informations fournies par la mesure de contribution, nous utiliserons les deux métrique $S$ et $D$. L'avantage de ces mesures, dont la signification est intuitive, est qu'elles offrent un ordre total entre différentes approximations. Cependant, elles nécessitent, pour choisir le point de référence, la définition d'une borne supérieure de la région dans laquelle se trouvent tous les points réalisables. Ce choix peut avoir un impact sur l'ordre entre les approximations. D'où l'intérêt d'utiliser ces mesures avec la contribution, et non seule.

Les métriques $S$ et $D$ permettent de comparer directement plusieurs ensembles de solutions, et de quantifier le rapport d'aire de dominance entre les fronts Pareto.

Lors de nos évaluations de performance ultérieures, nous prendrons comme point de référence le point *Nadir* (le vecteur des pires valeurs pour chaque objectif) de l'ensemble des ensembles Pareto comparés.

### 4.5.5 Guimoo : Une interface graphique pour les problèmes d'optimisation multi-objectifs

Guimoo (a Graphical User Interface for Multi Objective Optimization) est un logiciel développé au sein de notre équipe. C'est un logiciel open source présentant une interface graphique pour les problèmes d'optimisation multi-objectif. Il offre une visualisation 2D et 3D des fronts de problèmes d'optimisation multi-objectifs. et permet aussi l'évaluation de performances des algorithmes d'optimisation multi-objectif. En effet l'évaluation n'est pas triviale, puisque ceux-ci fournissent un ensemble de solutions non dominées (front Pareto) et donc non-comparables entre elles. En particulier la comparaison d'algorithmes peut être difficile. Il est donc nécessaire d'utiliser plusieurs mesures de performance spécifiques afin d'évaluer ces algorithmes.

Guimoo est composé de trois fenêtres principales :
- la fenêtre principale permettant la gestion du problème, la visualisation des fronts, le calcule des métriques etc.
- la fenêtre *PO* files* qui affiche la listes des fronts ouverts.
- la fenêtre *Contents of PO* files* affiche le contenu de chaque front.

Dans la figure 4.9, nous pouvons voir l'interface globale de GUIMOO ainsi que l'af-

fichage de 3D des quatre fronts pareto trouvés par exécution de notre algorithme muti-objectif avec des données biopuce.

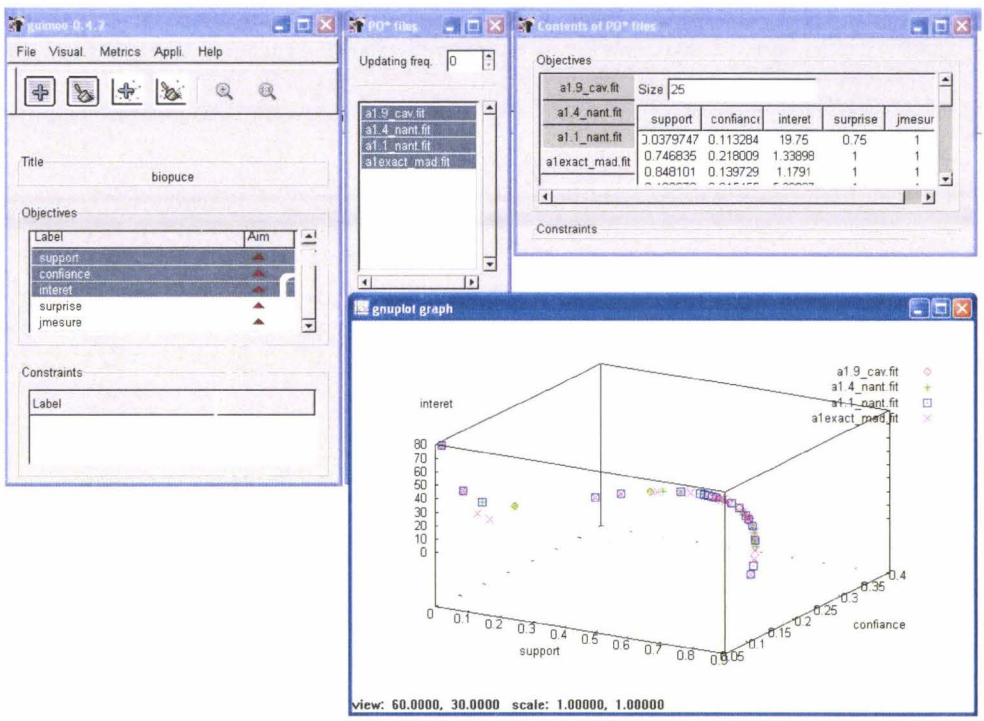



FIG. 4.9 – Guimoo : front.

Guimoo intègre un ensemble varié de mesures, affiche le résultat du calcul et offre la possibilité d'exporter le résultat dans plusieurs formats (excel, latex, csv, etc) (voir la figure 4.10). Les mesures intégrées sont :

- Métrique S (S-metric)
- Métrique D (D-metric)
- Métriques R (R-metrics)
- Contribution
- Entropie (entropy)
- Distance générationnelle (generational distance)
- Spacing
- Couverture (coverage)
- Différence de couverture (coverage difference)

La figure 4.10 montre deux boites de calcul pour deux mesures : la contribution (la proportion de solutions non-dominées entre les fronts deux à deux) et la S-metric (calcule la taille de l'espace des objectifs que le front domine).

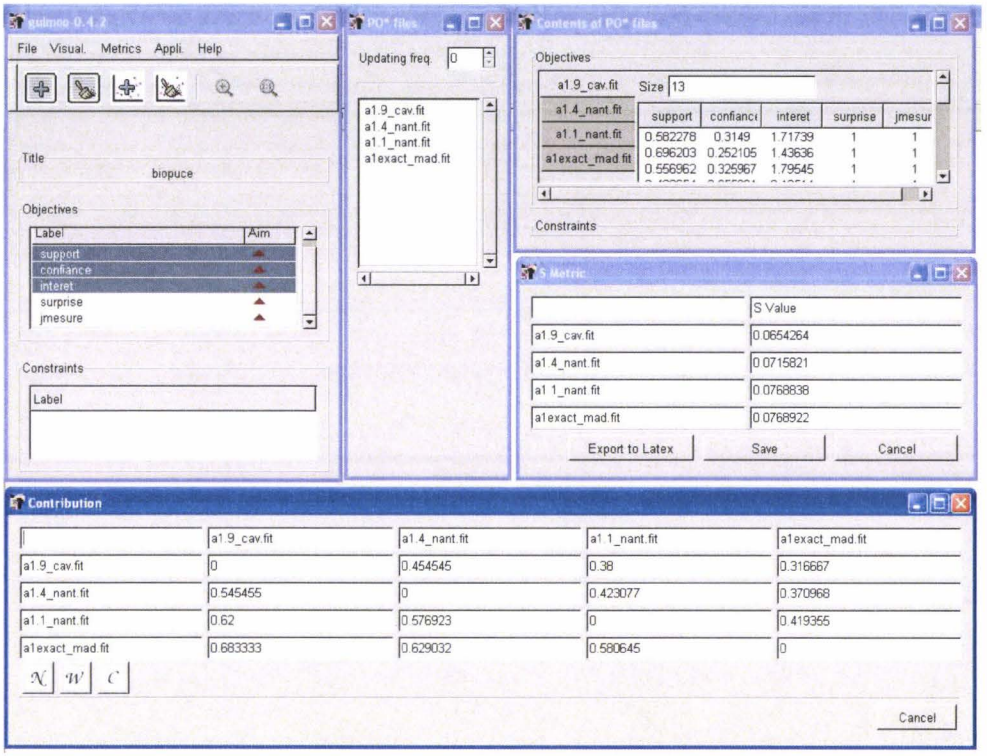



FIG. 4.10 – Guimoo : métrique.

## 4.6 Conclusion

Nous avons présenté dans ce chapitre les principaux concepts de l'optimisation multi-objectif et les différentes approches de résolution pour traiter un problème d'optimisation mono ou multi-objectif : les méthodes exactes et les méthodes approchées. Ainsi, nous avons présenté brièvement les métaheuristiques les plus connues pour l'optimisation mono et multi-objectif. Récemment, beaucoup de recherches ont été menées sur l'application des algorithmes évolutionnaires aux problèmes d'optimisation multi-objectif. Celles-ci ont permis de mettre en avant l'intérêt d'utiliser des méthodes d'optimisation basées sur le concept de population. Deb et Goel [DG01a] illustrent leurs originalités et leurs bonnes performances sur de nombreuses instances de problèmes.

Dans la troisième partie, nous avons présenté un état de l'art sur les différentes mesures d'évaluation de la qualité d'un front Pareto. En effet, l'évaluation des résultats d'un algorithme mutli-objectif est un problème délicat qui oblige à utiliser plusieurs mesures différentes, car il est impossible de représenter par une seule valeur la qualité des solutions, la taille du front et la répartition des solutions sur le front.

Pour le reste de cette thèse, nous nous plaçons dans le cadre de problèmes d'optimisation où il n'existe pas de modèle de préférences sur les critères (tous les critères sont de même importance).

Nous avons déjà présenté dans le deuxième chapitre les méthodes classiques (la famille

fichage de 3D des quatre fronts pareto trouvés par exécution de notre algorithme muti-objectif avec des données biopuce.

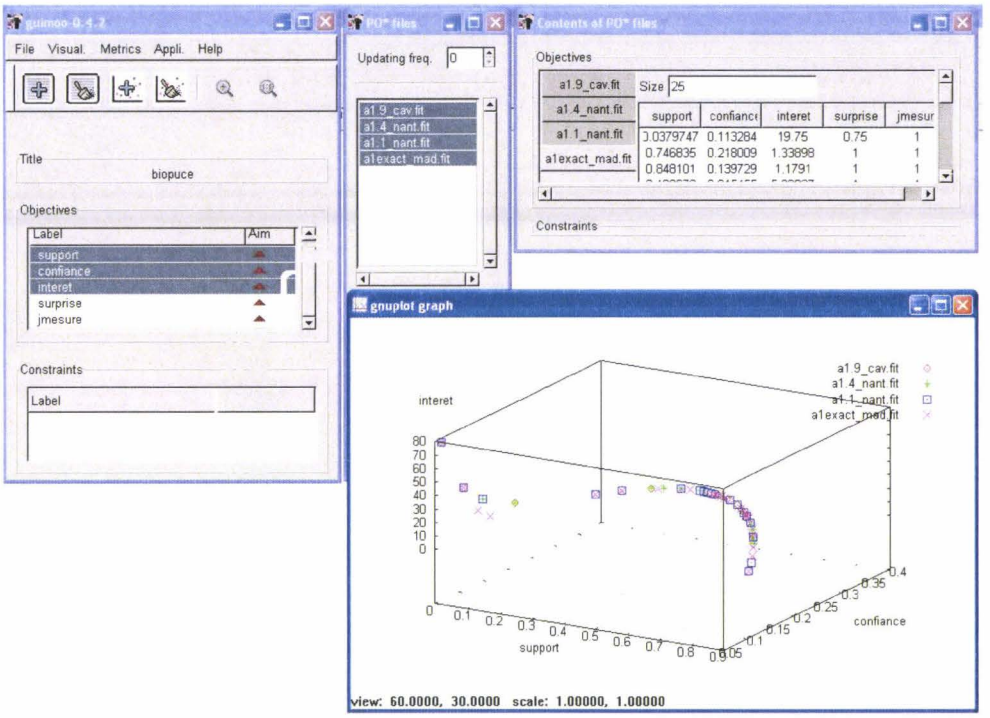



FIG. 4.9 – Guimoo : front.

Guimoo intègre un ensemble varié de mesures, affiche le résultat du calcul et offre la possibilité d'exporter le résultat dans plusieurs formats (excel, latex, csv, etc) (voir la figure 4.10). Les mesures intégrées sont :

- Métrique S (S-metric)
- Métrique D (D-metric)
- Métriques R (R-metrics)
- Contribution
- Entropie (entropy)
- Distance générationnelle (generational distance)
- Spacing
- Couverture (coverage)
- Différence de couverture (coverage difference)

La figure 4.10 montre deux boites de calcul pour deux mesures : la contribution (la proportion de solutions non-dominées entre les fronts deux à deux) et la S-metric (calcule la taille de l'espace des objectifs que le front domine).

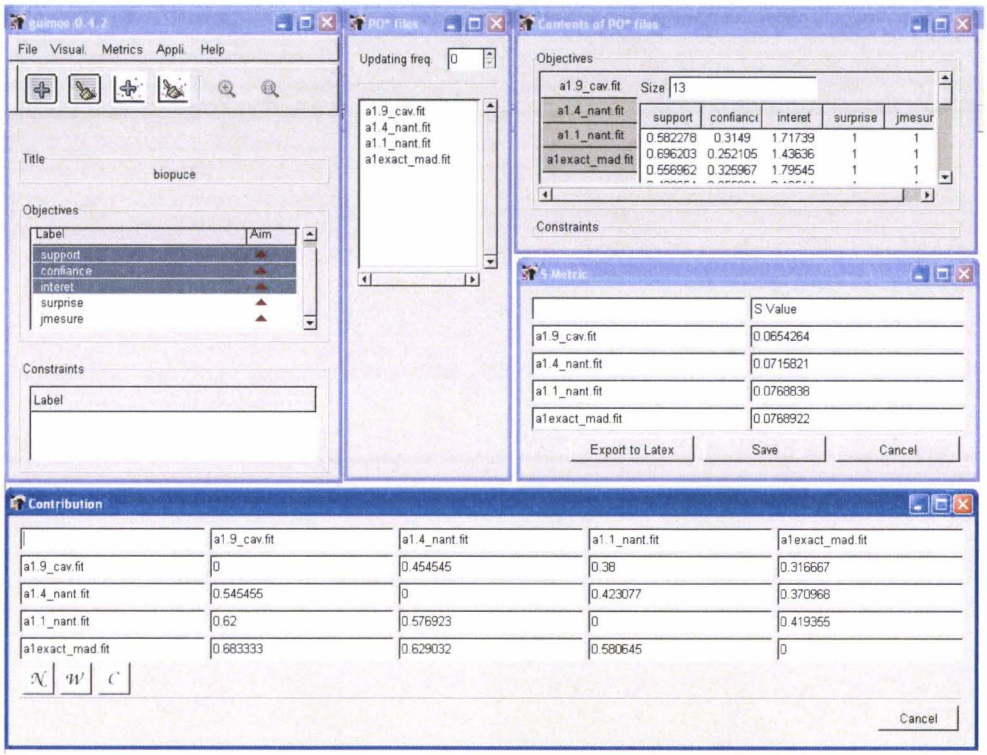



FIG. 4.10 – Guimoo : métrique.

## 4.6 Conclusion

Nous avons présenté dans ce chapitre les principaux concepts de l'optimisation multi-objectif et les différentes approches de résolution pour traiter un problème d'optimisation mono ou multi-objectif : les méthodes exactes et les méthodes approchées. Ainsi, nous avons présenté brièvement les métaheuristiques les plus connues pour l'optimisation mono et multi-objectif. Récemment, beaucoup de recherches ont été menées sur l'application des algorithmes évolutionnaires aux problèmes d'optimisation multi-objectif. Celles-ci ont permis de mettre en avant l'intérêt d'utiliser des méthodes d'optimisation basées sur le concept de population. Deb et Goel [DG01a] illustrent leurs originalités et leurs bonnes performances sur de nombreuses instances de problèmes.

Dans la troisième partie, nous avons présenté un état de l'art sur les différentes mesures d'évaluation de la qualité d'un front Pareto. En effet, l'évaluation des résultats d'un algorithme mutli-objectif est un problème délicat qui oblige à utiliser plusieurs mesures différentes, car il est impossible de représenter par une seule valeur la qualité des solutions, la taille du front et la répartition des solutions sur le front.

Pour le reste de cette thèse, nous nous plaçons dans le cadre de problèmes d'optimisation où il n'existe pas de modèle de préférences sur les critères (tous les critères sont de même importance).

Nous avons déjà présenté dans le deuxième chapitre les méthodes classiques (la famille

des algorithmes Apriori) de résolution pour le problème des règles d'association. Ces méthodes ont montré leur efficacité pour trouver des solutions satisfaisantes pour un grand nombre de problèmes de petite taille et pour deux critères (le support et la confiance). Cependant, ces méthodes ne sont pas adaptables pour une résolution multi-objectif. Le plus gros problème concerne l'impossibilité de mesurer la qualité d'une solution sans utiliser le support et la confiance et d'introduire les mécanismes multi-objectif.

Nous allons proposer dans le chapitre suivant une approche de résolution multi-objectif pour les règles d'association en utilisant plusieurs critères (voir chapitre 3) avec un algorithme évolutionnaire muti-objectif.

# Chapitre 5

# Algorithmes génétiques pour les règles d'association multi-objectifs

Sommaire


5.1 Algorithme génétique pour les règles d'association . . . . . 89
5.2 Mécanismes et opérateurs multi-objectifs . . . . . 94
5.3 Implémentation . . . . . 97
5.4 Résultats expérimentaux . . . . . 103
5.5 Conclusion . . . . . 107


Afin de pouvoir traiter le problème de recherche de règles avec plusieurs critères nous avons développé un algorithme génétique multi-objectif adapté pour la recherche de règles en utilisant les cinq objectifs choisis suite à une analyse statistique réalisée et exposée au chapitre 3. Il s'agit donc d'optimiser le Support, la Confiance, la J-mesure, l'Intérêt et la Surprise. Un codage et des opérateurs spécifiques pour les règles ainsi que des mécanismes de recherche multi-objectifs sont implémentés (mutation adaptative, sélection Pareto, élitisme et archive Pareto). Ce travail a fait l'objet de présentations aux conférences ROADEF 2003 [KDT03a] et CEC 2004 [KDT04b].

## 5.1 Algorithme génétique pour les règles d'association

Un algorithme génétique est un algorithme itératif de recherche globale dont le but est d'optimiser une ou plusieurs fonctions d'évaluation. Pour atteindre ce but l'algorithme travaille en parallèle sur une population de solutions candidates appelées individus ou chromosomes. Chaque individu est constitué d'un ensemble d'éléments appelés caractéristiques ou gènes pouvant prendre plusieurs valeurs appartenant à un alphabet non nécessairement numérique. Le but est donc de trouver la meilleure combinaison de ces éléments afin d'atteindre le maximum d'adéquation. A chaque itération, appelée génération, est créée une nouvelle population d'individus. Cette nouvelle génération est constituée généralement par

des individus mieux adaptés à l'environnement (désignés par la fonction d'évaluation). Ces individus sont créés en utilisant des parties des meilleurs éléments de la population précédente ainsi que des parties novatrices. Au fur et à mesure des générations, les individus vont tendre vers de "bonnes" solutions.

L'objectif du travail présenté dans ce chapitre est de concevoir et de développer un algorithme génétique adapté aux problèmes de recherche de règles d'association sous une plate forme générique (EO : voir la section 5.3.1). Nous présentons dans cette partie les opérateurs génétiques spécifiques au problème de recherche des règles d'association.

### 5.1.1 Codage et représentation des solutions

Les AGs travaillent sur une population d'individus. Chaque individu est composé d'un ensemble d'éléments appelés gènes pouvant prendre plusieurs valeurs (allèles).

Il existe deux approches différentes pour extraire des règles en utilisant un AG : l'approche de Pittsburgh [Smi83] et l'approche de Michigan [Hol75b]. La première consiste à coder plusieurs règles au sein d'un même individu tandis que dans la seconde, une règle ne code qu'un seul individu.

Dans l'AG que nous mettons en œuvre dans la thèse, chaque individu représente une règle de la forme **IF C THEN P** (figure 5.1), où C est la Condition et P est la conséquence (Prédiction). La condition est une conjonction de termes de la forme suivante : $terme_1$ and $terme_2$ and ... and $terme_N$ où *and* est l'opérateur logique AND et **N** est la longueur maximum de la partie condition choisie par l'utilisateur. Chaque terme est un triplet : <attribut,opérateur,valeur>, où l'opérateur représente une expression logique (< ,<= ,> ,<= ,=) ou une expression relationnelle dépendante du type d'attribut. La conséquence dans le cas général peut contenir également une conjonction de termes. Nous nous placerons dans un cas particulier où la conséquence ne sera composé que d'un seul terme.

Dans le cas des donnés biopuces, un *attribut* est un gène et *valeur* représente les différentes expressions géniques du gène (I, D, MI, MD, NC dans le cas des puces Affymetrix).

### 5.1.2 Génération de la population initiale

La population initiale représente un ensemble d'individus de taille POPSIZE. Chaque individu est une solution potentielle du problème de taille maximale VECSIZE fixée par l'utilisateur, et limitée par la taille des données du problème. La population est générée aléatoirement, ce qui permettra d'avoir une bonne répartition des individus dans l'espace de recherche. Les doublons (individus codant une même règle) sont éliminés et regénérés (Voir l'algorithme 8).

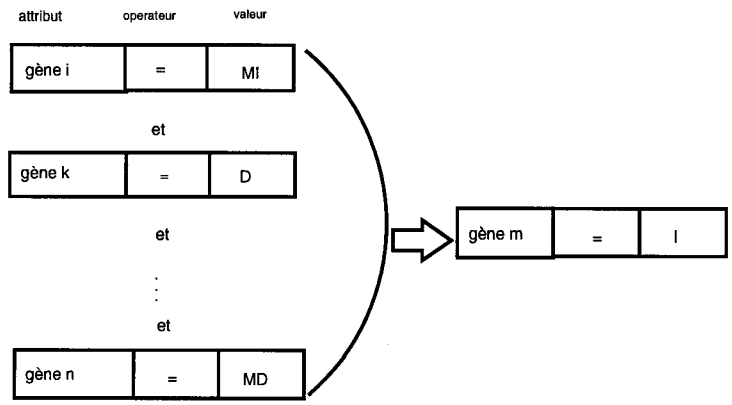



FIG. 5.1 – Codage de la solution.

**Algorithme 8** Pseudo-code du générateur de la population initiale

**Entrées:** $POPSIZE, VECSIZE$
**Sorties:** $POP$

```
igeno ← 0
tantque igeno < POPSIZE faire
  x ← random(VECSIZE) ;
  solution ← generate_solution_randomly(taille : x) ;
  evaluation(solution) ;
  si solution ∉ POP alors
    add(solution, POP) ;
    igeno ← igeno + 1 ;
  finsi
fin tantque
Retourner POP.
```

### 5.1.3 L'opérateur de croisement

Le croisement (crossover en anglais) est une opération binaire qui s'applique sur deux individus parents choisis au hasard dans la population d'individus déjà sélectionnés. Le rôle de cet opérateur est la recombinaison des caractéristiques de deux individus, assimilés aux parents, afin d'obtenir deux individus (enfants). Le croisement est appliqué selon une probabilité de croisement choisie par l'utilisateur.

Soient R1=IF C1 THEN P1 et R2=IF C2 THEN P2 deux règles (individus) parents. Nous proposons deux types de croisement à appliquer en fonction des attributs communs existants ou non entre les règles : Croisement par mutation de valeurs et Croisement par insertion d'attributs.

#### 5.1.3.1 Croisement par mutation de valeurs

Si les deux parties conditions des deux règles ont un ou plusieurs attributs communs, l'un de ces attributs est choisit aléatoirement et ses valeurs correspondantes dans les deux parents sont permutées (Voir la figure 5.2).

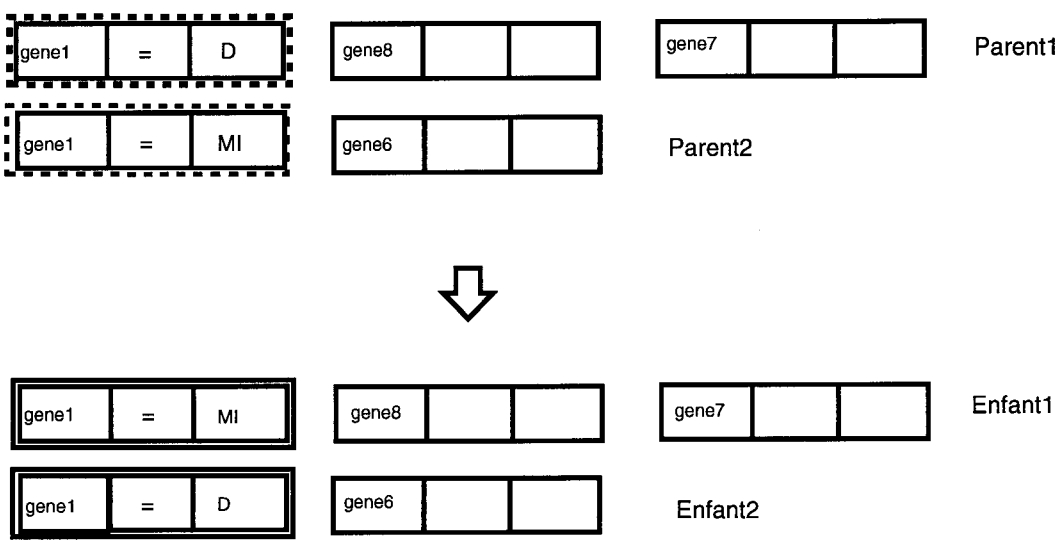



FIG. 5.2 – Croisement par mutation de valeurs.

#### 5.1.3.2 Croisement par insertion d'attributs

Si les deux parties conditions des deux règles n'ont pas d'attribut commun, on insère un terme de C1 choisi aléatoirement dans C2 avec une probabilité $Pi$ :

$$Pi = \frac{(MAXTERM - K)}{MAXTERM} \tag{5.1}$$

Où K est le nombre d'attributs dans C1 (ou C2 dans le second cas) et MAXTERM est le nombre maximum d'attributs autorisés. On applique la même procédure pour insérer un terme de C2 dans C1. La figure 5.3 illustre le fonctionnement de cet opérateur.

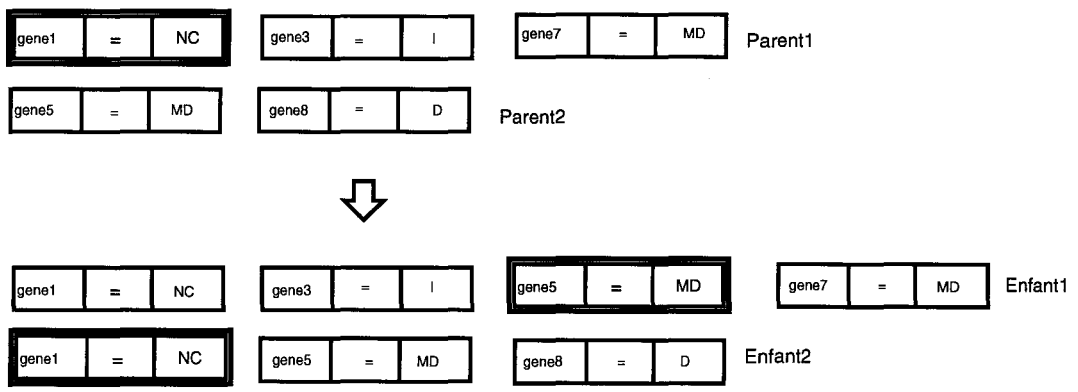



FIG. 5.3 – Croisement par insertion d'attributs.

### 5.1.4 L'opérateur de mutation

L'opérateur de mutation agit sur un individu. Il consiste à choisir un ou plusieurs attributs au hasard dans un individu et à modifier leurs valeurs de manière aléatoire. Quatre opérateurs de mutation ont été mis en œuvre :

- ***Mutation par valeur*** : l'opérateur choisit aléatoirement un attribut et en modifie sa valeur en choisissant aléatoirement une autre valeur du domaine. Par exemple, pour la règle : *IF (gène1=I) and (gène5=I) THEN (gène2=D)*, la mutation peut choisir aléatoirement l'attribut *gène1* et la valeur *MD*. La nouvelle règle obtenue est alors : *IF (gène1=**MD**) and (gène5=I) THEN (gène2=D).*

- ***Mutation par attribut*** : l'opérateur choisit aléatoirement un terme et remplace son attribut par un autre déterminé aléatoirement. La valeur du nouvel attribut est choisie aléatoirement dans son domaine. Par exemple, considérons la règle : *(gène1=I) and (gène5=I) THEN (gène2=D)*, la mutation choisit aléatoirement le terme *(gène5=I)* et le remplace par *(gène1000=I)*. La nouvelle règle obtenue est alors : *IF (gène1=I) and (**gène1000**=I) THEN (gène2=D).*

- ***Mutation par insertion*** : l'opérateur ajoute un nouveau terme à une règle. Le terme est choisi aléatoirement dans son domaine. Par exemple, pour la règle : *IF (gène1=I) THEN (gène2=D)*, la mutation peut choisir aléatoirement le terme *(gène5=MI)*. La nouvelle règle obtenue est alors : *IF (gène1=I) and **(gène5=MI)** THEN (gène2=D).*

- ***Mutation par suppression*** : l'opérateur supprime un terme de la règle (si le nombre de termes est suffisant). Par exemple, pour la règle : *IF (gène1=I) and*

*(gène5=MI) THEN (gène2=D)*, la mutation peut choisir aléatoirement le terme *(gène5=MI)*. La nouvelle règle obtenue est alors : *IF (gène1=I) THEN (gène2=D)*.

### 5.1.5 La mutation adaptative

Nous avons quatre opérateurs de mutation : la mutation par valeur, la mutation par attribut, la mutation par suppression et la mutation par insertion. Fixer les probabilités de mutation lorsque l'on a plusieurs opérateurs est difficile et souvent réalisé de façon expérimentale. Nous avons voulu pallier à ce problème en mettant en place un calcul adaptatif du taux d'application de chaque opérateur de mutation en fonction de l'amélioration des solutions qu'il apporte. Plus un opérateur est efficace, plus il sera utilisé. De nombreux auteurs ont mené des expérimentations sur l'adaptation des probabilités d'application des opérateurs. Dans [HWC00], les auteurs proposent de calculer ce nouveau taux de mutation en évaluant le progrès d'une mutation $M_i$ pour un individu *ind* muté en un individu *mut* : Dans le cas mono-objectif, le progrès (progress) est calculé de cette manière :

$$progress(M_i) = Max(fitness(ind), fitness(mut)) - fitness(ind)$$

Ensuite pour toutes les mutations, on calcule le ratio des gains relativement à $Nb\_mut(M_i)$ le nombre de mutations effectuées par l'opérateur $M_i$ :

$$Gain(M_i) = \frac{progress(M_i)/Nb\_mut(M_i)}{\sum_j \left(progress(M_j)/Nb\_mut(M_j)\right)}$$

On fixe un taux de mutation minimum $\delta$ et un taux de mutation global $p_{mutation}$ pour $N$ opérateurs de mutation à appliquer. On obtient pour le calcul des nouveaux taux de mutation :

$$p(M_i) = Gain(M_i) \times (p_{mutation} - N \times \delta) + \delta$$

La somme des taux de mutation est égale au taux de mutation $p_{mutation}$. Le taux de mutation initial est fixé à $p_{mutation}/N$.

Dans le cas multi-objectif, il faut redéfinir le calcul de la valeur du progrès en utilisant la notion du Pareto dominance. Il s'agit d'affecter un progrès de 1 à l'opérateur de mutation $M_i$ lorsque celui-ci, appliqué à une solution *ind*, permet d'engendrer une solution mutée *mut* dominant *ind*. Dans le cas où *ind* domine *mut*, le progrès vaudra 0. Dans les autres cas, lorsque la solution *mut* est non comparable avec *ind*, les progrès prend 1/2 [Bas05].

## 5.2 Mécanismes et opérateurs multi-objectifs

L'adaptation des algorithmes évolutionnaires pour la résolution de problèmes multi-objectifs porte d'une part, sur l'étape d'évaluation des individus de manière à prendre en compte les différents critères à optimiser et d'autre part, sur les étapes de sélection et de remplacement en maintenant l'ordre partiel défini par la relation de dominance.

L'objectif de ces mécanismes est de favoriser la recherche de solutions non dominées tout en conservant une diversité suffisante.

### 5.2.1 La sélection

Comme son nom l'indique, la sélection vise à sélectionner une sous population qui serviront à la reproduction à partir d'une population parent. La méthode la plus courante est celle initiée par Holland [Hol75b] en 1975 : la sélection par roulette, qui est une méthode de sélection proportionnelle au niveau de fitness des individus. Le nombre de fois qu'un individus sera sélectionné est égal à son fitness divisé par la moyenne du fitness de la population totale. Cette fonction est déterminante dans un algorithme génétique et de nombreuses méthodes de sélection bien plus complexes sont disponibles : la sélection par rang, la sélection par tournoi...

Dans notre cadre d'étude multi-objectif, la phase de sélection favorise les individus les mieux adaptés avec plusieurs fonctions objectif pour participer à la phase de reproduction. Elle se base sur le ranking et la sélection par roulette. En effet l'utilisation d'une sélection basée sur la notion de dominance de Pareto va faire converger la population vers un ensemble de solutions de meilleurs compromis. Plus le rang d'un individu est petit, plus sa probabilité de sélection augmente.

Nous avons utilisé et comparé deux méthodes : le ranking Pareto [FF93] et le ranking calculé comme proposé dans NSGA [DAPM00]. Pour le ranking Pareto, chaque individu de la population est rangé en fonction du nombre d'individus qui le dominent (voir figure 5.4). Nous rappelons que la procedure NSGA attribue un rang égal à 1 pour tous les individus non dominés de la population courante qui forment le front de Pareto $R1$. La méthode procède en suite récursivement en attribuant le rang $k$, aux individus non dominés de la population initiale de laquelle ont été retirés les individus des rangs 1 à k-1. Ce processus récursif s'arrête lorsqu'un rang unique a été associé à tous les individus de la population courante (voir figure 5.5).

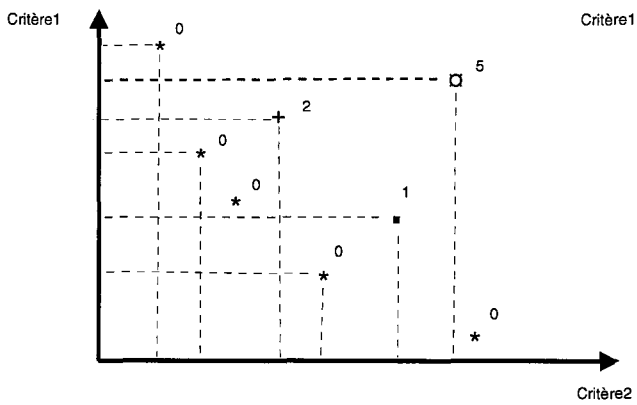



FIG. 5.4 – Pareto ranking.

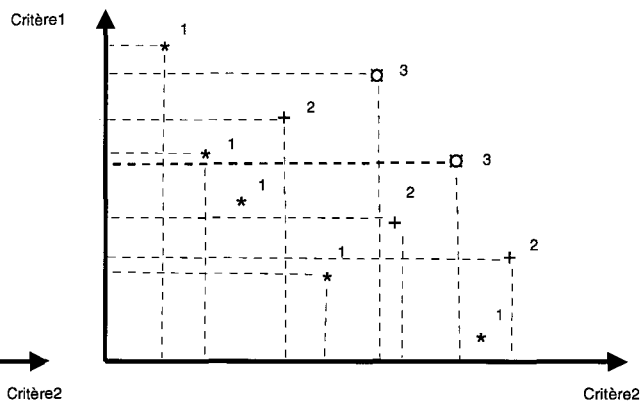



FIG. 5.5 – NSGA ranking.

### 5.2.2 Elitisme et archive Pareto

A la création d'une nouvelle population, il y a de grandes chances que les meilleures solutions soient perdues après les opérations de croisement et de mutation. Pour éviter cela, on utilise la méthode d'élitisme. Elle consiste à copier une ou plusieurs des meilleures solutions dans la nouvelle génération. Ensuite, on génère le reste de la population selon l'algorithme de reproduction usuel. Cette méthode améliore considérablement les algorithmes génétiques, car elle permet de ne pas perdre les meilleurs solutions.

Une des premières implémentations de ce mécanisme dans un algorithme génétique est présentée dans [Jon75]. L'élitisme est introduit pour conserver les bonnes solutions lors du passage de la génération courante à la génération suivante. Conserver ces solutions pour les générations futures permet d'améliorer les performances des algorithmes sur certains problèmes.

Réaliser un algorithme élitiste dans le cadre des problèmes multi-objectif est plus difficile que pour les problèmes mono-objectif. En effet, la meilleure solution n'est plus une solution unique, mais tout un ensemble. Deux adaptations du mécanisme élitiste sont considérées : la première approche regroupe les algorithmes fondés sur les travaux de De Jong, qui conservent pour les générations futures les $k$ meilleures solutions [TMA+95, DAPM00]. Mais comment, avec cette approche, sélectionner $k$ solutions, si l'ensemble des solutions non dominées comporte plus de $k$ solutions ? Il y a un risque de perdre une partie du front Pareto optimal, et le concept de l'élitisme n'est plus complètement présent. Les approches récentes [IMT95, PM98a, ZT98] tendent à utiliser une population externe d'individus dans laquelle est stocké le meilleur ensemble des solutions non dominées découverts jusqu'ici. Cet ensemble est mis à jour continuellement pendant la recherche, et les solutions stockées continuent à pouvoir être choisies par l'opérateur de sélection. Ils peuvent ainsi se reproduire et transmettre leurs caractéristiques aux générations suivantes.

Actuellement, les algorithmes élitistes obtiennent de meilleurs résultats sur un grand nombre de problèmes multi-objectifs [ZT99, DG01b].

#### 5.2.2.1 La sélection élitiste

Ainsi, dans le domaine de l'optimisation multi-objectif, la sélection élitiste consiste à maintenir la seconde population Archive Pareto, contenant les solutions non dominées trouvées au cours des différentes générations de l'algorithme évolutionnaire. Les individus de cette population participent avec une certaine probabilité à l'étape de sélection et donc à la reproduction de nouveaux individus.

#### 5.2.2.2 Remplacement élitiste

Le remplacement consiste à remplacer les plus mauvaises solutions par les nouvelles solutions générées par les opérateurs de croisement et de mutation qui les dominent. La taille de la population reste inchangée. La procedure de remplacement est :

```
Replacement(population, offspring, size)
```

```
{
  Add offspring to population;
  remove the redundant rules;
  Pareto ranking (population);
  Truncate(population, size);
}
```

La figure 5.6 présente le schéma général de l'algorithme génétique multi-objectif.

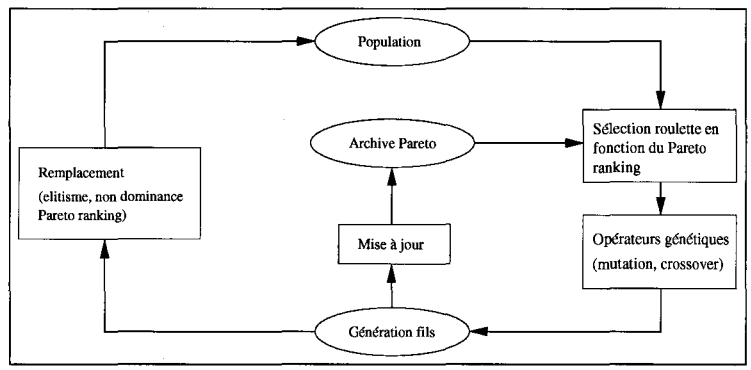



FIG. 5.6 – L'algorithme génétique multi-objectif.

## 5.3 Implémentation

L'algorithme proposé à été implémenté sous la la librairie d'évolution artificielle EO (Evolving Object).

### 5.3.1 EO : plateforme de développement

EOlib est une bibliothèque *Open Source*, développée en C++, dédiée à la conception d'applications à base d'algorithmes évolutionnaires. Ses principales caractéristiques sont les suivantes : elle est libre de tout paradigme, très flexible par son mécanisme d'encapsulation des opérateurs. De nombreuses facilités sont par ailleurs fournies, telles que la possibilité de définition de paramètres en ligne, l'analyse et la visualisation des résultats, ainsi que des mécanismes de points de reprise, ...EO a été appliquée à de nombreux domaines : l'optimisation de perceptrons multi-couches, la segmentation de voix et d'images, l'ingénierie en industrie automobile, ...

La plateforme EO [KMRS01] répond à quatre objectifs principaux :
- Réutilisabilité : plusieurs mécanismes peuvent être réutilisés.
- Extensibilité : gain du temps en développement et la possibilité d'intégration de nouveaux composants.

- Flexibilité : décomposition quasi-atomique des tâches à effectuer.
- Adaptabilité : la possibilité d'intégrer ses propres composants permet de combiner de mécanismes génériques avec des mécanismes spécifiques au problème traité.

Le schéma de la figure 5.7 synthétise le fonctionnement d'un AG et les principales entités qui le composent.

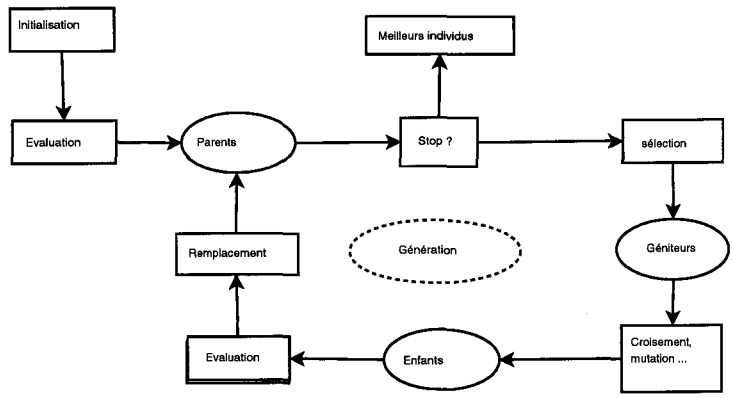



FIG. 5.7 – Fonctionnement d'un algorithme génétique.

A partir des principales étapes constituant le schéma d'exécution itératif d'un AG. (Fig. 5.7) ont été identifiés les principaux composants (Fig. 5.8) associés aux opérateurs génétiques mis en oeuvre. On distinguera les mécanismes dont le fonctionnement est indépendant du problème traité (la sélection « eoSelect » , le remplacement « eoReplacement » , le critère de décision de continuation « eoContinue » ) des autres plus spécifiques (la génération d'une solution initiale aléatoire « eoInit » , la fonction d'évaluation « eoEvalFunc » , les opérateurs de variation « eoQuadOp » et « eoMonOp » ), etc.

Une métrique d'évaluation doit être définie, associant à une solution donnée plusieurs mesures de qualité (cas multi-objectif). Pour notre problème de recherche des règles d'association, la qualité d'une règle est associée à plusieurs valeurs réelles (le Support, la Confiance, la J-mesure, l'Intérêt et la Surprise). Les opérateurs, croisement et mutation sont aussi dépendants de notre problème traité. En général, on distinguera les mécanismes dont le fonctionnement est indépendant du problème traité (la sélection, le remplacement, le critère d'arrêt) des autres mécanismes plus spécifiques (la génération aléatoire des solutions, la fonction d'évaluation, l'opérateur de croisement et l'opérateur de mutation).

Notons que EO intègre de nombreuses fonctionalités et services facilitant son exploitation : une visualisation des résultats, le traitement en ligne de paramètres, des mécanismes pour la sauvegarde ou reprise de l'état d'exécution (*checkpointing*), la génération automatique de statistiques, ...

Enfin, EO a été couplée avec le logiciel EASEA [CLSL00]. Ce dernier interprète un langage de spécification de haut niveau, dans lequel on modélise à la fois le problème d'optimisation et les opérateurs de transformation, puis génère du code EO directement

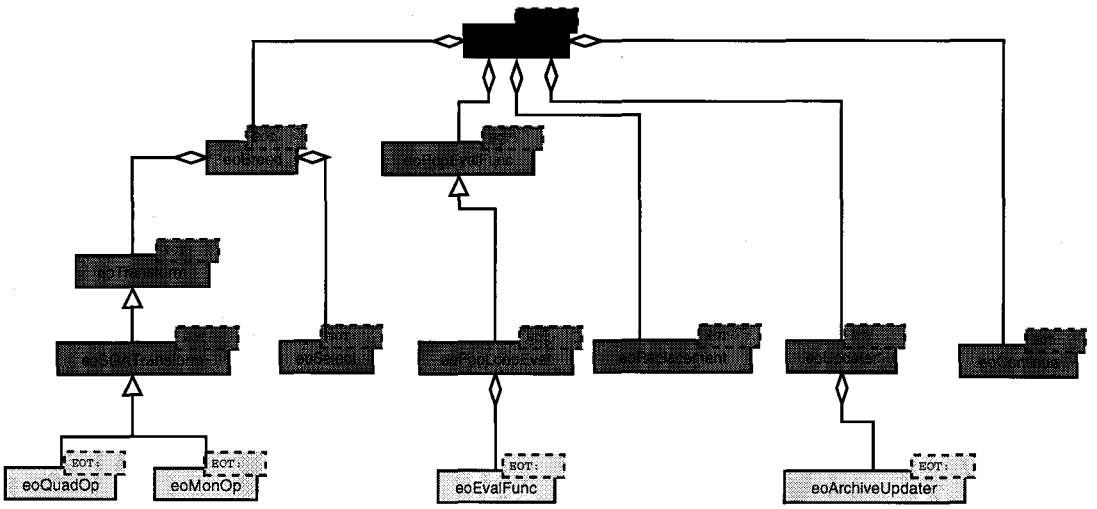



FIG. 5.8 – Principaux composants de EO.

compilable. Ainsi, la plate-forme EO peut être classée « boîte noire » .

Dans notre implémentation, nous générons un ensemble de règles d'associations aléatoires de type « RULE » afin de créer la population initiale « POP » .

Les composants génériques sont facilement intégrables dans le programme (voir code dans la figure 5.9) par le fait qu'ils requièrent le paramètre « RULE » . Il s'agit des opérateurs de sélection dans la population « eoSelectOne » , de composition d'un ensemble de parents « eoSelect » , de composition d'une nouvelle génération « eoReplacement » , de décision de continuation du processus d'évolution « eoContinue » , etc. Chacune de ces interfaces est implémentée par un ensemble de classes correspondantes pour chacune à une stratégie donnée.

Au contraire, les composants devant être implémentés sont dépendants de la représentation adoptée. Il s'agit de l'opérateur d'initialisation de solutions aléatoires « eoInit » , de la fonction d'évaluation « eoEvalFunc » , de l'opérateur de croisement « eoQuadOp » et le l'opérateur de mutation « eoMonOp » .

L'ensemble de composants instanciés et nécessaires à la composition de notre algorithme évolutionnaire est résumé dans le tableau 5.1.

### 5.3.2 MOEO : Optimisation Multi-objectif avec EO

La plate-forme EO autorise l'optimisation multi-objectif. Des classes spécifiques sont dédiées à une représentation multi-critère des valeurs de qualité. Il conviendra préalablement de définir le nombre d'objectifs à prendre en compte, et leurs « buts » respectifs.

Diverses techniques ont été déjà intégrées dans la plate-forme pour une approche Pareto dans la résolution de problèmes d'optimisation multi-critères. Cette approche se base directement sur la notion de dominance dans la sélection des solutions générées. Dans cette

| Opérateurs génériques | |
|---|---|
| eoEasyEA | algorithme évolutionnaire |
| eoGenContinue ou eoSteadyFitContinue | critère d'arrêt |
| eoRouletteWorthSelect | sélection par roulette |
| eoSelectNumberFromPopAndArch | élitisme |
| eoParetoRanking | Pareto ranking |
| eoNDPlusReplacement | remplacement -dominance- |
| eoAdaptCombinedMonOp | mutation adaptative |
| eoCheckPoint | Check Point |
| eoArchiveUpdater | MAJ de l'archive |
| **Opérateurs spécifiques** | |
| RuleEval | fonction d'évalation |
| RuleInit | générateur aléatoire de règles |
| Crossover | opérateur de croisement |
| Mut_Value, Mut_Att, Mut_Del, Mut_Add | opérateurs de mutations |

TAB. 5.1 – Principaux composants génériques et spécifiques nécessaires à la composition de l'algorithme évolutionnaire pour la recherche de règles d'association.

```
int main (int __argc, char * * __argv)
 {
 eoArchive<RULE>archive; init_pareto(); RuleInit init; RuleEval eval;
 eoPop <RULE>pop(POP_SIZE, init);

/*Mutation and Crossover*/

 Crossover _cross; Mut_att _mutate1; Mut_value _mutate2; Mut_add
_mutate3; Mut_del _mutate4 ;

/* Combined  mutation */
 eoAdaptCombinedMonOp<RULE>_combined_mut(_mutate1,eval,0.25,0.05);
 _combined_mut.add (_mutate2,0.25);
 _combined_mut.add(_mutate3,0.25);
 _combined_mut.add(_mutate4,0.25);

 eoUpdateMuteRate<RULE>Rate_update(_combined_mut,nbregenerationMAJ);
 eoSGATransform<RULE>transform(_cross,CROSS_RATE,_combined_mut,MUT_RATE);

/* selection Pareto */
 eoDominanceMap <RULE> dom_map ;
 eoParetoRanking <RULE> pareto_rank (dom_map) ;
 eoRouletteWorthSelect <RULE> select1 (pareto_rank) ;
 eoSelectNumber <RULE> selectPareto (select1,T_SIZE) ; /* selection Elitism */
 eoRandomSelect <RULE> arch_select ;
 eoSelectNumberFromPopAndArch<RULE>selectEP(select1,arch_select,arch,ProbSelectPop,T_SIZE);
 eoNDPlusReplacement<RULE> replace(pareto_rank);/* Replacement*/

 eoSteadyFitContinue<RULE> cont( MAX_GEN, 10);
 eoCheckPoint<RULE>checkpoint(cont) ;
 eoArchiveUpdater<RULE>arch_updater(arch) ;
 checkpoint.add(arch_updater) ;

 eoEasyEA<RULE>ea(checkpoint,eval,selectElitismPareto,transform,replace);
 ea(pop)
}
```

FIG. 5.9 – Extrait du code.

intention ont été intégrées des méthodes de « ranking » (*e.g.* NSGA [SD95b], NDS [FF95a], ...), dont le rôle est d'établir un rang entre les individus. Il s'agit également de générer des solutions diversifiées dans l'espace des critères. Les méthodes de maintien de la diversité (*e.g.* la fonction de partage ou « Sharing » [GR87b]), par la formation de niches écologiques et d'espèces apparaissent particulièrement utiles pour stabiliser des sous-populations multiples le long de la frontière Pareto. Le lecteur est invité à consulter [Tal00] pour une présentation détaillée des techniques introduites précédemment.

La plupart des efforts employés aux problèmes multi-objectifs au sein de « EO » se sont concentrés sur l'étape de sélection. Aussi, nos contributions concernent essentiellement l'élitisme, l'ajout d'opérateurs génétiques adaptatifs et enfin l'intégration de métriques pour l'évaluation des performances.

#### 5.3.2.1 Elitisme

Cette technique consiste à maintenir une population autre que la population courante et archivant toutes les solutions Pareto optimales générées au cours de la recherche. La figure 5.10 montre la mise en oeuvre immédiate d'un tel mécanisme. En effet, la comparaison au sens Pareto d'une solution à une autre est indépendante du problème, mais relève uniquement de la relation de non-dominance aux objectifs.

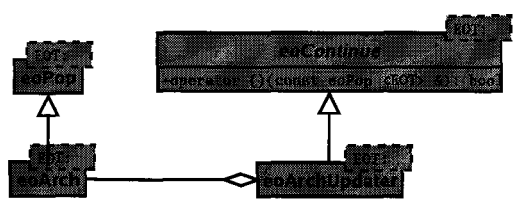



FIG. 5.10 – Gestion d'une archive de solutions Pareto Optimales générées durant la recherche.

Cette population externe (archive), actualisée à chaque itération, participe dans la définition des opérateurs génétiques (sélection, reproduction, ...). L'élistisme est très utilisé dans le processus de sélection. Il consiste par exemple à réaliser la sélection des individus aussi bien dans la population courante que des solutions non dominées trouvées pendant la recherche.

L'implémentation du modèle élitiste (Fig. 5.11) requiert la définition des stratégies quant aux sélections respectives depuis la population courante et de l'archive.

Cette archive de solutions Pareto optimales est aussi exploitée dans le modèle parallèle en îles (voir le chapitre 6 où des échanges de solutions s'effectuent entre des archives distribuées). Il conviendra de définir des politiques de migration pertinentes, et tout particulièrement dans le processus de sélection des solutions à émettre.

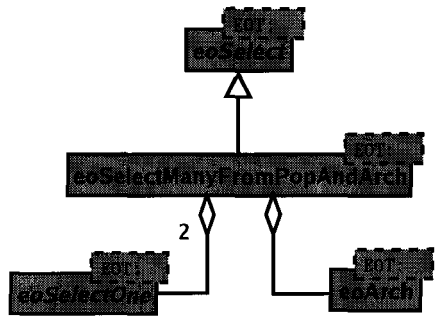



FIG. 5.11 – Un opérateur de sélection combiné composant un ensemble de parents à partir de la population courante et de l'élite.

#### 5.3.2.2 Opérateurs de diversification

Les techniques de diversification (Sharing) doivent être utilisées afin d'assurer une bonne diversité des solutions générées. La diversification a pour objectif le maintien de la diversité au sein d'une population d'individus. La qualité d'un individu est dégradée par rapport au nombre d'individus similaires dans la population et à leur distribution.

Une formalisation de cet opérateur est présentée dans [Tal00]. Elle est intégrée dans « MOEO » (Fig. 5.12) en tant que métrique « eoPerfToWorth » appliquée à transformer un vecteur de valeurs aux différents objectifs en une valeur scalaire de performance. Dans son fonctionnement, remarquons que cet opérateur se base sur la distance euclidienne normalisée des solutions d'une population.

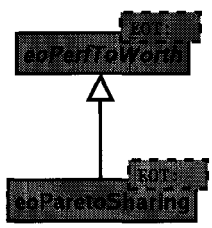



FIG. 5.12 – Un opérateur pour la caractérisation des solutions parmi les plus diversifiées dans l'espace des critères.

#### 5.3.2.3 Opérateurs adaptatifs

La mise en œuvre d'opérateurs de croisement et de mutation adaptatifs a permis l'obtention de meilleurs résultats. Le principe est de concevoir des opérateurs de variation encapsulant d'autres opérateurs de croisement ou de mutation. A l'activation d'un tel composant « combiné» , la probabilité de sélection d'un opérateur interne est directement liée à sa performance récente (*i.e.* a t-il contribué à faire progresser le front de solutions Pareto-optimales ?). Ces stratégies ont été intégrées dans « MOEO » . Elles se définissent comme des opérateurs de recombinaison ou de mutation encapsulant chacun un ensemble

d'opérateurs du même type (Fig. 5.13).

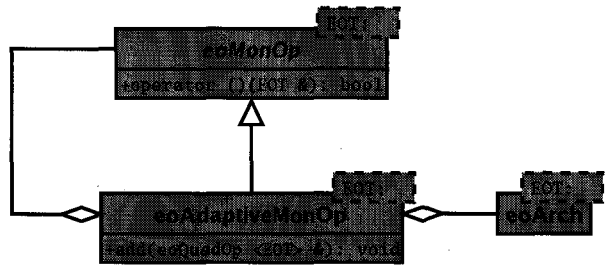



FIG. 5.13 – Une représentation UML de l'opérateur de mutation adaptatif.

## 5.4 Résultats expérimentaux

Les résultats sont obtenus à partir de plusieurs exécutions du programme sous un/-plusieurs PC(S) de type Pentium IV Intel 3 Ghz, équipé d'une mémoire DDR de 512 Mo.

Tout au long de ces tests, nous considérons la qualité de l'ensemble des solutions Pareto comme facteur de performance. Lors de ces expérimentations, nous avons fait en premier lieu plusieurs exécutions sur des instances des bases de données classiquement utilisées en data mining *UCI Machine Learning Repository* [NHBM98], ce qui nous a permis de valider l'approche de la résolution et la robustesse des algorithmes génétiques.

### 5.4.1 Bases de données

Nous avons expérimenté cet algorithme sur plusieurs bases de données : deux bases de données de l'UCI *Nursery School et Connect-4 opening* et deux bases relatives à des expérimentations sur puces à ADN (BD1 et YeastBD).

**Nursery School :**

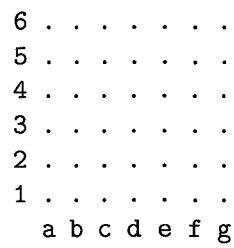
Cette base publique a été utilisée par différents auteurs afin d'évaluer leurs approches [Fre99, ALF99]. La base *Nursery School* permet la classification de candidats à une école maternelle. Elle est formée de 12 960 instances et de 9 attributs dont le dernier attribut est l'attribut but pour les règles de classification cherchées (voir tableau 5.2).

**Connect-4 opening :**

Il s'agit du jeu de Puissance 4. Le premier à aligner 4 pions horizontalement, verticalement ou en diagonale a gagné sur un jeu 6*7. La base de données contient 67557 instances et 42 attributs, chaque attribut correspond à une position de connect-4 et il y a trois valeurs $x$, $o$ et $b$ (x=player x has taken, o=player o has taken, b=blank). Le dernier attribut(43) est classe win, loss, draw. L'attribut *classe* est l'attribut but pour les règles d'association.
La carte est numérotée de la façon suivante :

| | Nom de l'attribut | Valeurs de l'attribut |
|---|---|---|
| **1** | parents | usual, pretentious, great_pret |
| **2** | has_nurs | proper, less_proper, improper, critical, very_crit |
| **3** | form | complete, completed, incomplete, foster |
| **4** | children | 1, 2, 3, more |
| **5** | housing | convenient, less_conv, critical |
| **6** | finance | convenient, inconv |
| **7** | social | nonprob, slightly_prob, problematic |
| **8** | health | recommended, priority, not_recom |
| **9** | recommendation | not_recom, recommend, very_recom, priority, spec_prior |

TAB. 5.2 – Structure de la base de données *Nursery School.*




**Bases de bonnées de puces à ADN :**

**DB1 :** Il s'agit d'une base de données confidentielle obtenue auprès de l'Institut Biologique de Lille (IBL) sur les maladies d'Alzheimer. Avec le vieillissement de la population, la maladie d'Alzheimer va devenir une priorité de santé publique. 600 000 personnes de plus de 75 ans sont atteintes et l'on dénombre près de 200 000 nouveaux cas par an. Cette base contient 22376 gènes (attributs) pour 45 puces à ADN de type Affymetrix. Chaque gène a cinq valeurs selon le traitement statistique d'Affymetrix : Increase (I), Marginal Increase (MI), lorsque le gène est sur exprimé, Decrease (D) et Marginal Decrease (MD) s'il est sous exprimé et No Change (NC) si la différence d'expression est non significative.

**YeastDB** "MIPS Yeast Genome Database" est une base de données publique contenant 2467 gènes pour 79 puces.
Le but est de trouver la relation entre gènes. La forme des règles : *IF (gène1=$val_1$) and (gène5=$val_2$) and ... (gèneN=$val_l$) THEN (gèneM=$val_k$)* où $val_i$ représente les différentes expressions géniques du gène. Par exemple, la règle : *IF (gène1=I) and (gène5=I) THEN (gène2=D).*

### 5.4.2 Analyse des opérateurs

La mise en place d'un tel algorithme a toujours son lot de questions quant aux opérateurs efficaces et aux mécanismes à mettre en œuvre tels que le mécanisme d'adaptativité des opérateurs, l'élitisme et l'archive Pareto.

Afin de comparer les fronts Pareto obtenus par différentes configurations, la mesure de la contribution présentée dans le chapitre précédent est utilisée. Rappelons qu'une contribution supérieure à 0.5 indique une amélioration du front.

#### 5.4.2.1 Stabilité de l'algorithme

Le tableau 5.3 donne les valeurs d'une analyse statistique descriptive faite sur le nombre des solutions obtenues après 10 exécutions. Cette analyse donne la valeur du minimum, du maximum et de la moyenne de la taille du front Pareto. Cette analyse statistique nous

| Dataset | Minimum | Maximum | Moyenne |
|---|---|---|---|
| Nursery dataset | 11 | 13 | 12.7 |
| Connect-4 dataset | 96 | 98 | 97.2 |

TAB. 5.3 – Nombre de solutions Pareto trouvées pour deux bases de données classiques de l'UCI.

montre que l'algorithme est relativement robuste quant à la cardinalité du front trouvé . En effet, le nombre de solutions Pareto trouvées est plus ou moins semblable d'une exécution à l'autre.

#### 5.4.2.2 Adaptativité des mutations

Quatre opérateurs de mutation ont été mis en œuvre pour former la version adaptative : la mutation par valeur, la mutation par attribut, la mutation par suppression et la mutation par insertion.

Deux versions ont été implémentées : la version dite non adaptative en utilisant la mutation par valeur et par attribut et la mutation adaptative en mettant en place un calcul du taux d'application de chaque opérateur de mutation en fonction de l'amélioration des solutions qu'il apporte. Plus un opérateur est efficace, plus il sera utilisé.

Le tableau 5.4 montre la contribution de la version adaptative (pour les deux bases de données d'expression génique). Tout d'abord le nombre de solutions Pareto obtenues est plus important avec la version adaptative ce qui permet donc une meilleure approximation de l'ensemble du front. De plus, la contribution supérieure à 0.5 indique que les fronts sont meilleurs (puisqu'ils dominent en partie les fronts obtenus sans l'adaptativité).

| | DB1 | YeastDB |
|---|---|---|
| Non adaptive | 9 sol. | 19 sol. |
| Adaptive | 12 sol. | 31 sol. |
| Contribution adaptative/non adaptative | 0.54 | 0.71 |

TAB. 5.4 – Adaptative vs non adaptative.

On peut expliquer ces résultats avec les deux figures 5.14 et 5.15 qui représentent les améliorations éventuellement obtenues à génération cela permet de visualiser la contribution de la mutation adaptative / non adaptative au fur et à mesure des générations. Il apparaît que la mutation adaptative continue à apporter des améliorations même lorsque l'AG a commencé à converger.

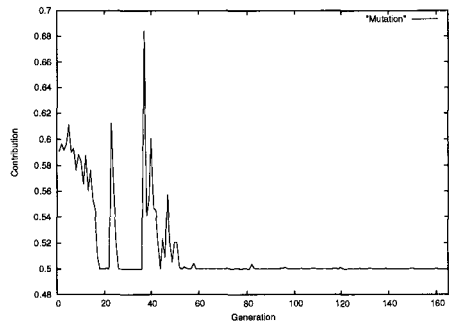



FIG. 5.14 – Contribution de la mutation non adaptive.

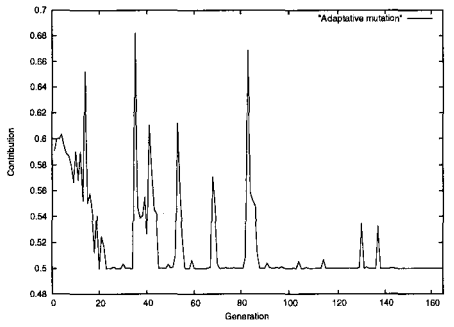



FIG. 5.15 – Contribution de la mutation adaptive.

#### 5.4.2.3 L'élitisme

Le tableau 5.5 montre la contribution de l'élitisme pendant la phase de sélection qui permet d'obtenir au final de meilleurs fronts. Notons que ceci est vérifé que ce soit pour une sélection de type Pareto Ranking ou une sélection de type NSGA.

| Contribution | Pareto without elitism | NSGA without elitism |
|---|---|---|
| Pareto with elitism | 0.64 | |
| NSGA with elitism | | 0.75 |

TAB. 5.5 – La contribution de l'élitisme (DB1).

Afin de choisir quelle est la méthode de ranking à utiliser, nous avons comparé les deux méthodes de ranking sur les bases de données de type génomique. Le tableau 5.6 montre que l'efficacité est semblable avec un petit avantage pour la procédure NSGA.

#### 5.4.2.4 Projections 2D

Un autre aspect intéressant à regarder est la structure des fronts Pareto obtenus. Pour cela, les sous-figures 5.16 présentent des projections suivant différents couples d'objectifs d'un front obtenu. Rappelons ici que les objectifs sont à maximiser. Ces figures montrent

| Contribution | Pareto élitisme | NSGA élitisme |
|---|---|---|
| Pareto élitisme | | 0.54 |
| NSGA élitisme | 0.46 | |

TAB. 5.6 – Contribution : Pareto ranking / NSGA (DB1).

que les objectifs choisis sont bien complémentaires, puisque les fronts comportent plusieurs solutions de compromis. Cela valide en partie l'analyse statistique faite pour la sélection des objectifs.

D'autre part, la figure 5.17 montre le front Pareto projeté suivant deux critères corrélés : Jmesure et Piatetsky-Shapiro (les deux mesures appartiennent au même groupe). Ce front contient une seule solution Pareto ce que confirme la forte corrélation entre ces deux critères (une solution optimisant l'un des critères optimise également le second).

### 5.4.3 Résultats : Règles d'association

Les tests concernant l'analyse des données génomiques ont été réalisés sur un PC 3Ghz sous Linux en utilisant la mutation adaptative et l'élitisme avec le ranking Pareto.

Les paramètres de l'algorithme génétique sont :

- Taille de la population : DB1 = 200, YeastDB = 1000
- Probabilité de sélection Pareto archive (élitisme) : 0.5
- Probabilité de Crossover : 0.8
- Probabilité de Mutation : 0.4
- Nombre de générations : 200

Les deux figures 5.18 et 5.19 présentent la projection suivant deux couples de critères (Surprise/Intérêt et Intérêt/Support) d'un front obtenu.

Le tableau 5.7 montre quelques règles lors d'une exécution de l'algorithme sur la base de données **DB1**. De plus, nous indiquons leurs valeurs d'évaluation dans le tableau 5.8 pour les cinq critères (Support, J-mesure, Intérêt, Surprise et Confiance).

## 5.5 Conclusion

Le problème de recherche de règles d'association peut se définir comme un problème d'optimisation combinatoire multi-objectif. Nous avons présenté dans ce chapitre un algorithme évolutionaire pour la recherche de règles d'association. Il permet de traiter des bases de données classiquement utilisées en data mining (*UCI Machine Learning Repository*), ainsi que des bases de données relatives à des expérimentations sur puces à ADN. Nous avons défini le codage et les différents opérateurs spécifiques pour les règles d'association, et des mécanismes multi-objectif ont été implémentés. Nous avons mis en place un mécanisme adaptatif pour pouvoir appliquer plusieurs mutations selon l'évolution de

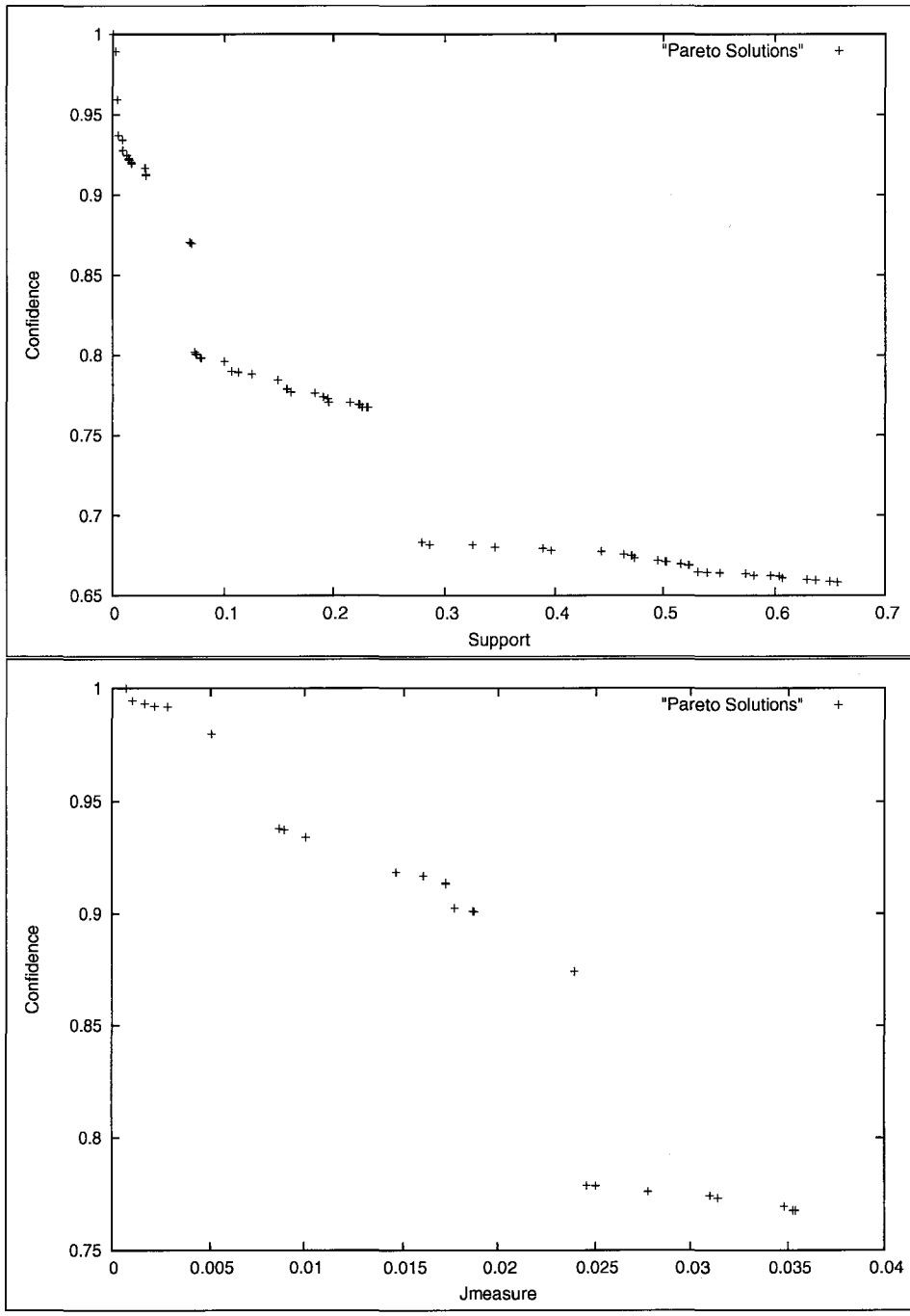



FIG. 5.16 – Projections 2D de l'ensemble des solutions du front Pareto avec des critères complémentaires sur la base de données publique de l'UCI connect-4.

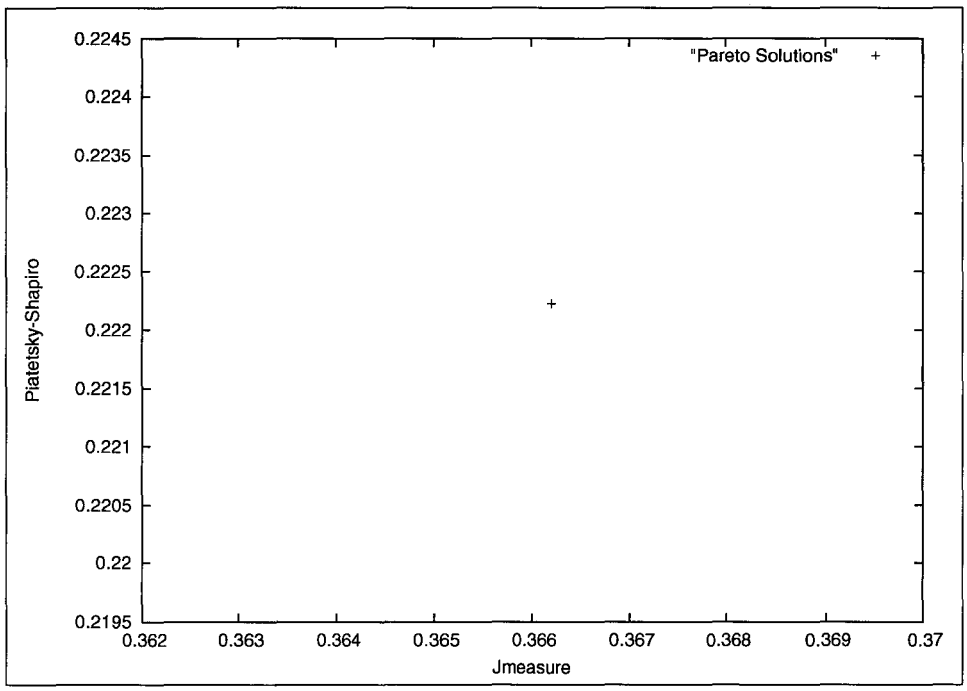



FIG. 5.17 – Projection 2D de l'ensemble des solutions du front Pareto avec deux critères fortement corrélés.

| Règles | Description |
|---|---|
| R1 | if ((122=I) and (372=NC) and (499=NC) and (435=NC)) then (222=I) |
| R2 | if ((436=NC) and (63=I) and (487=NC) and (332=NC) and (210=I) and (374=NC)) then (219=I) |
| R3 | if ((161=I) and (229=I) and (118=D) and (503=NC) and (311=NC)) then (39=D) |
| R4 | if ((238=I) and (226=I) and (426=NC))) then (64=I) |
| R5 | if((146=I) and (318=NC) and (499=NC) and (435=NC) and (457=NC) and (479=NC) and (367=NC) and (457=NC)) then (222=I) |

TAB. 5.7 – Description de quelques solutions Pareto obtenus sur la base de données biologique (DB1).

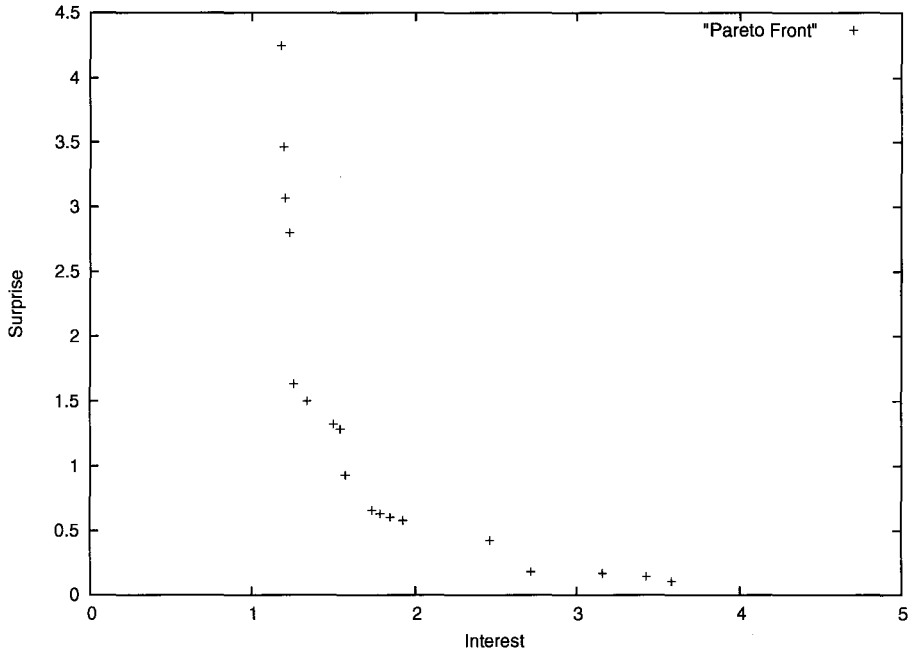



FIG. 5.18 – Front Pareto (Surprise/Intérêt) - YeastDB.

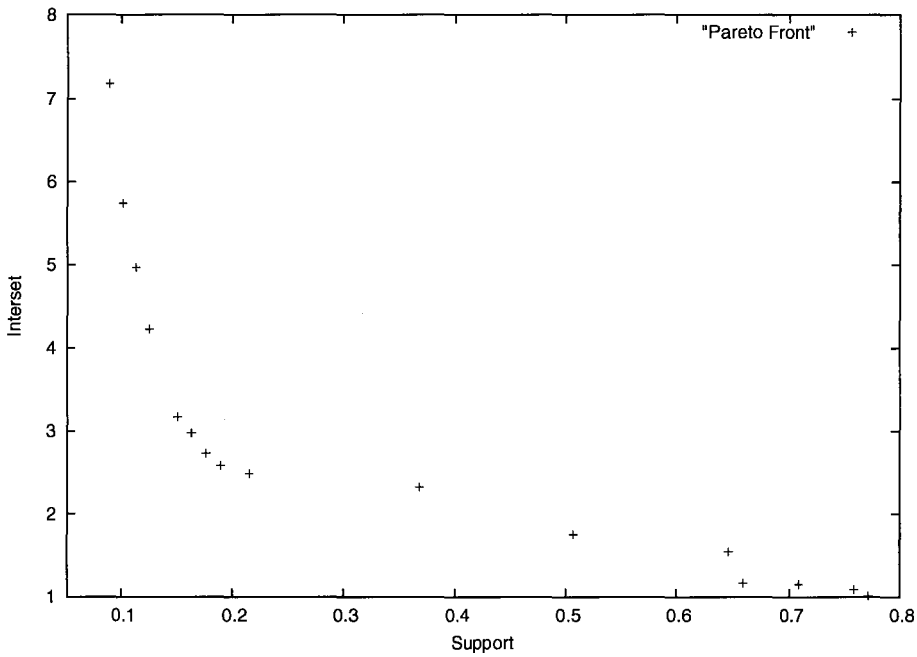



FIG. 5.19 – Front Pareto (Intérêt/Support) - YeastDB.

| Règles | Support | J-mesure | Intérêt | Surprise | Confiance |
|---|---|---|---|---|---|
| R1 | 0.133 | 0.248 | 6.428 | 0.857 | 1.000 |
| R2 | 0.155 | 0.268 | 5.625 | 0.857 | 0.875 |
| R3 | 0.244 | 0.268 | 3.000 | 0.733 | 1.000 |
| R4 | 0.244 | 0.268 | 3.000 | 0.733 | 1.000 |
| R5 | 0.688 | 0.153 | 1.250 | 0.838 | 0.861 |

TAB. 5.8 – La qualité de solutions Pareto obtenues sur la base de données biologique (DB1) (présentées dans 5.7).

l'algorithme et adapter leur taux d'application en fonction de l'amélioration apportée par chacun d'eux.

Afin de rendre l'algorithme plus performant (qualité et diversité), nous allons présenter, dans le prochain chapitre, une approche coopérative en utilisant la plateforme PARADISEO [Cah05]. Le but de cette approche est de combiner différentes metaheuristiques et/ou méthodes exactes afin d'améliorer la qualité des solutions Pareto obtenues.

# Chapitre 6

# Méthodes coopératives pour les règles d'association multi-objectif

Sommaire


6.1 Introduction . . . . . 113
6.2 Approche parallèle . . . . . 114
6.3 Approche hybride . . . . . 127
6.4 Applications : intégration du module . . . . . 133
6.5 Conclusion . . . . . 142


## 6.1 Introduction

Dans les chapitres précédents nous avons discuté de la modélisation multi-objectif du problème de recherche de règles d'association et nous avons proposé une première approche de résolution utilisant les algorithmes génétiques. Étant donné le très large espace de recherche relatif à ce type de problème, une question se pose quant à l'efficacité absolue de la méthode proposée et il semble intéressant de regarder ce que pourrait apporter une approche coopérative (parallélisme, hybridation).

Deux types d'approches sont considérées dans ce chapitre. Elles concernent soit la parallélisation de méthodes de même type (entre AG), soit l'hybridation de méthodes différentes. Ainsi, nous présentons dans un premier temps une approche parallèle développée pour le problème de recherche de règles, dans laquelle différents algorithmes génétiques coopèrent [KDT05a, KDT05b]. Puis dans un deuxième temps nous présentons une approche hybride entre une métaheuristique (algorithme génétique) et une méthode exacte (un algorithme énumératif) en utilisant la plateforme PARADISEO [Cah05]. Ce travail a fait l'objet de deux chapitres dans deux livres "Parallel Computing for Bioinformatics and Computational Biology" [KDT06b] et "Handbook of Bioinspired Algorithms and Applications" [JKDT05].

De plus, afin de permettre une meilleure diffusion de notre approche, notre algorithme

génétique multi-objectif pour les règles d'association à été adapté et intégré dans le service de datamining relatif au projet GGM [1] et dans un pluging BASE [2].

## 6.2 Approche parallèle

L'intérêt de la parallélisation des algorithmes génétiques est de gagner en temps de calcul et d'améliorer la qualité des solutions. La méthode naturelle de lancement en parallèle des calculs de performance n'est pas la seule manière d'envisager la parallélisation des algorithmes génétiques. Il existe même une gamme complète de manières de paralléliser ces algorithmes, de la simple parallélisation du calcul de performance jusqu'à la distribution complète de la population sur les divers processeurs disponibles. Actuellement, les différentes architectures permettent de classer l'ensemble des algorithmes génétiques parallèles en trois types : le modèle centralisé, le modèle cellulaire et le modèle en îles (figure 6.1).

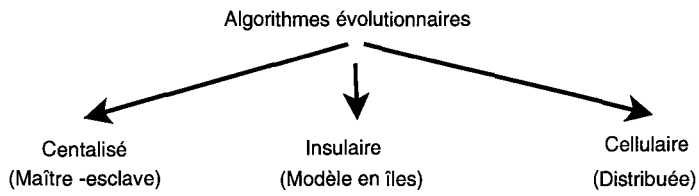



FIG. 6.1 – Classification des modèles parallèles.

### 6.2.1 Modèles parallèles pour les AGs

#### 6.2.1.1 Modèle centralisé : l'évaluation parallèle

L'opération la plus coûteuse dans les algorithmes évolutionnaires est généralement la phase d'évaluation des solutions. Ce modèle parallèle maintient la population totale sur une seule machine mais se sert des autres pour y confier les évaluations, afin qu'elles se fassent en même temps (modèle maître/esclave : figure 6.2).

#### 6.2.1.2 Modèle cellulaire : population distribuée

Le modèle de parallélisation cellulaire est un modèle totalement distribué selon un parallélisme à grain fin. Ce modèle consiste à distribuer une unique population sur l'ensemble des processeurs (en général sur des machines massivement parallèles). Sur chaque processeur alors quelques individus (souvent un seul) sont répartis, et les opérations de sélection/remplacement et de croisement se font entre individus "voisins" pour la topologie du réseau de processeurs (figure 6.3). Ce modèle est rarement utilisé aujourd'hui. En effet,

[1] Le projet GGM est un projet financé par l'ACI Masse de Données
[2] http ://base.thep.lu.se

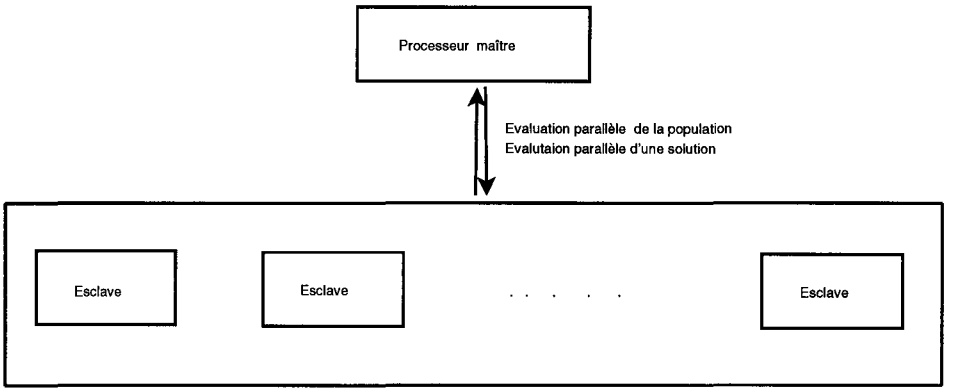



FIG. 6.2 – Modèle centralisé

le succès de ce modèle était lié au succès des machines massivement parallèles dans les années 90.

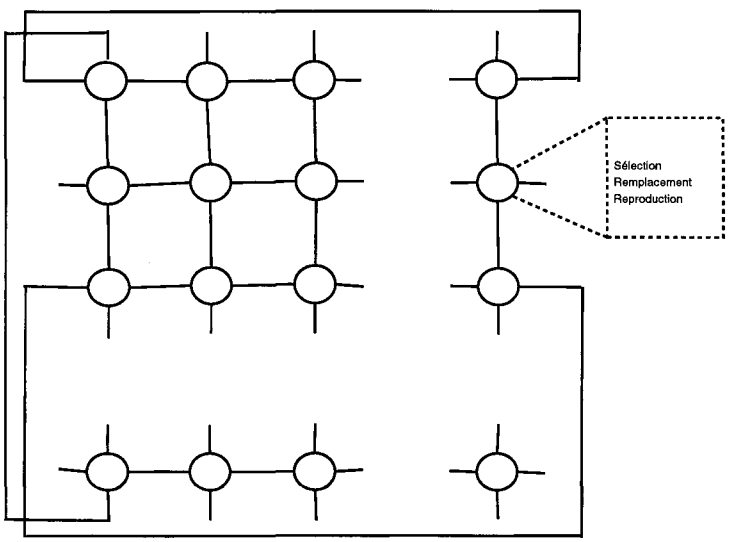



FIG. 6.3 – Modèle cellulaire.

#### 6.2.1.3 Modèle insulaire : Modèle en îles

Contrairement aux modèles parallèles précédents qui permettent essentiellement d'améliorer les temps de recherche, le modèle insulaire a pour objectif d'améliorer la qualité des résultats obtenus grâce à la coopération entre les AGs par migration de solutions. Le modèle insulaire consiste à diviser la population globale en plusieurs sous-populations (îles) et à les répartir sur l'ensemble des machines dont on dispose. Chaque sous-population ou île envoie ses meilleurs individus vers les populations (îles) voisines et reçoit ensuite des

individus envoyés par ses voisins suivant une topologie donnée.

Dans la suite de ce chapitre, nous avons utilisé ce modèle pour résoudre notre problème de recherche de règles d'association multi-objectifs en utilisant les algorithmes génétiques. Nous avons choisi ce modèle non seulement pour l'amélioration de l'efficacité de notre algorithme en terme de temps d'exécution, mais surtout de la qualité des fronts Pareto trouvés (la convergence et la diversité).

### 6.2.2 Algorithme génétique parallèle proposé

Nous proposons d'utiliser un modèle en îles (voir figure 6.4 ) où plusieurs AG définis au chapitre précédent sont déployés pour faire évoluer simultanément différentes populations (îles) de solutions et chaque envoie régulièrement quelques solutions de son archive Pareto locale à l'île voisine suivant la topologie en anneau.

Dans notre modèle insulaire, c'est une partie de l'archive Pareto qui est migrée. En effet, notre AG multi-objectif gère une autre population en parallèle de la population courante qui représente toutes les solutions Pareto trouvées durant la recherche. Une méthode de sélection élitiste est aussi utilisée dans chaque AG. Elle permet de sélectionner des individus de l'archive Pareto qui participent ainsi à la phase de reproduction d'individus et à l'intensification de la recherche. Les sous-populations associées aux AGs sont organisées selon une topologie en anneau sur laquelle va s'appliquer le mécanisme de migration. Le modèle insulaire que nous proposons est asynchrone ; aucune synchronisation n'est effectuée entre les sous-populations dans la phase de communication : pour un AG donné, lorsqu'un critère de migration est vérifié, l'AG concerné envoie une partie de son archive à l'AG voisin qui met à jour son archive locale. La prise en compte de la migration est évènementielle, c'est à dire que la réception n'est pas bloquante. Nous utilisons ici une fréquence de migration comme critère de migration, mais d'autres critères peuvent être utilisés comme par exemple la convergence de la sous-population courante. La figure 6.4 illustre la migration des archives Pareto entre quatre AGs organisés en anneau.

### 6.2.3 Politique d'échange

Pour chacune des îles, le processus de gestion des échanges asynchrones intervient au terme de chaque génération de l'AG, succédant à la phase de remplacement. La politique d'échange d'individus entre îles est définie par les paramètres suivants : le critère de décision d'échange, la topologie des échanges d'individus, le nombre d'émigrants intervenant dans une opération d'échange et la stratégie de leur sélection, la politique d'intégration des immigrants.

#### Topologie des échanges d'individus

La topologie des échanges d'individus indique pour chaque île ses voisins au regard de la migration i.e. l'île (ou les îles) de destination/provenance de ses emigrants/immigrants. Les modèles en anneau (figure 6.5) et hypercube (figure 6.6) sont d'ailleurs largement

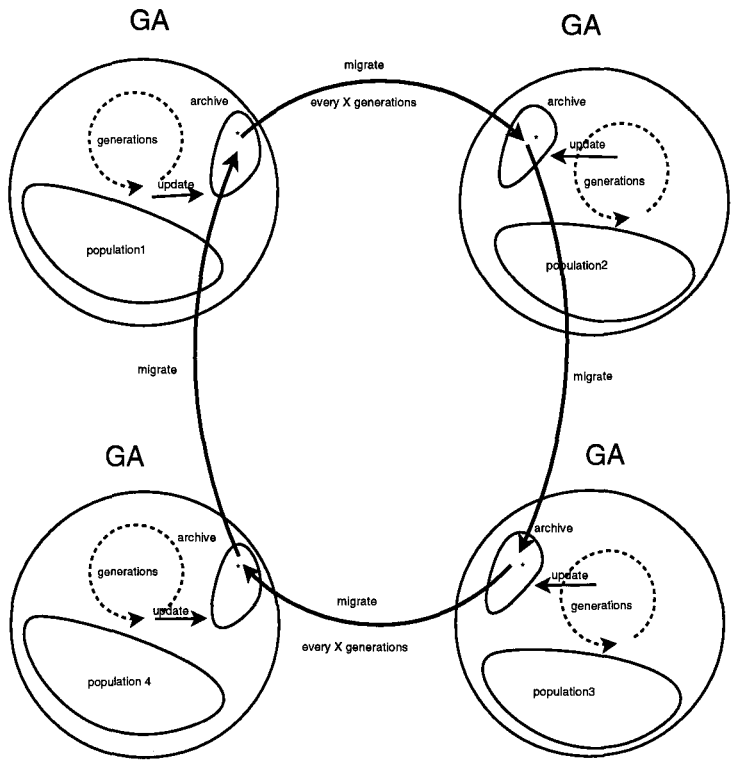



FIG. 6.4 – Algorithme génétique parallèle proposé : modèle en îles.

utilisés. II faut noter qu'un nombre trop élevé de jonctions entre îles se révèle inefficace car l'ensemble des populations distribuées se comportent alors comme une seule population globale.

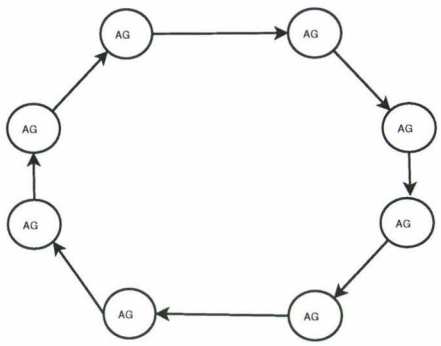



FIG. 6.5 – Topologie en anneau.

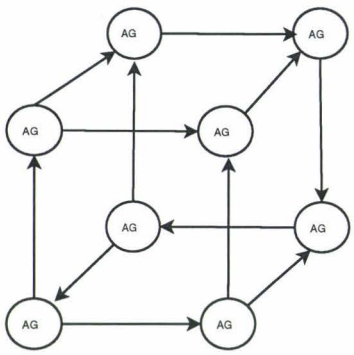



FIG. 6.6 – Topologie en hypercube.

### Politique de sélection des émigrants

La politique de sélection des émigrants indique pour chaque île, de manière élitiste ou aléatoire, les individus à migrer. La stratégie aléatoire ne garantit pas la sélection des meilleurs individus, mais elle a un coût de calcul plus faible. La stratégie élitiste, basée sur le rang Pareto, tente de sélectionner les meilleurs individus de la population (archive Pareto). Dans notre étude, les individus à migrer sont sélectionnés aléatoirement dans l'archive Pareto.

### Politique d'intégration des immigrants

De manière symétrique, la politique d'intégration des immigrants indique de manière élitiste ou aléatoire les individus de la population de destination à remplacer par les nouveaux arrivants. Dans notre cas, la politique consiste à la mises à jour de l'archive Pareto locale avec les solutions immigrantes.

### Expérimentations

Pour définir la meilleure politique d'échange, nous avons effectué différentes expérimentations afin de déterminer quand et combien de solutions doivent être envoyées. Pour ces expérimentations, une base de données d'expression génique est utilisée (yeastDB). Les paramètres par défaut sont :

- taille de la population = 300,
- sélection = 2/ 3 (200 ),
- taux de mutation = 0.5,

- taux de croisement = 0.8,
- sélection dans l'archive Pareto (élitisme) = 0.5,
- nombre minimal de générations = 300.

Les résultats représentent des moyennes sur un minimum de 10 exécutions. De nouveau, la contribution et l'entropie sont utilisées (Rappel : La contribution indique le ratio des solutions non dominées d'un front par rapport à un autre. Une contribution supérieure à 0.5 est une amélioration. L'entropie mesure la diversité du front. Plus l'entropie est proche de 1, mieux les solutions sont réparties sur le front).

### Nombre d'émigrants

Ce paramètre peut être défini comme un nombre fixe ou variable d'individus, ou comme un pourcentage d'individus de l'archive Pareto. Le choix de ce paramètre est crucial. En effet, s'il est trop faible les îles auront tendance à évoluer de manière indépendante et la migration aura moins d'impact sur le retard de la convergence et donc sur la qualité des solutions obtenues. Si le nombre d'emigrants est trop élevé le coût de communication sera plus important, et les îles auront tendance à converger vers les mêmes solutions. Un compromis est donc à trouver pour allier à la fois l'exploration et l'intensification de la recherche.

TAB. 6.1 – Comparaison de plusieurs scénarii sur le pourcentage de solutions échangées.

| | Contribution moyenne (10 exécutions) | | | | |
|---|---|---|---|---|---|
| | 2% | 7% | 10% | 20% | 50% |
| 2% | - | 0.47 | 0.47 | 0.51 | 0.51 |
| 7% | 0.53 | - | 0.48 | 0.54 | 0.54 |
| 10% | **0.53** | **0.52** | - | **0.54** | **0.56** |
| 20% | 0.49 | 0.46 | 0.46 | - | 0.50 |
| 50% | 0.49 | 0.46 | 0.44 | 0.50 | - |

Le tableau 6.1 indique les contributions deux à deux des fronts obtenus à l'aide de différents scénarii dans lesquels le pourcentage des solutions de l'archive Pareto envoyées à l'île voisine varie de 2% à 50 % . Il apparaît clairement ici que pour cette base de données, le pourcentage à 10 % est meilleur que tous les autres, comme le confirme la figure 6.7 qui représente la contribution moyenne d'une configuration par rapport à toutes les autres.

### Fréquence de migration

La migration d'individus d'une île vers une autre est périodique. Elle intervient sur chaque AG après un nombre de générations fixé par l'utilisateur (fréquence de migration).

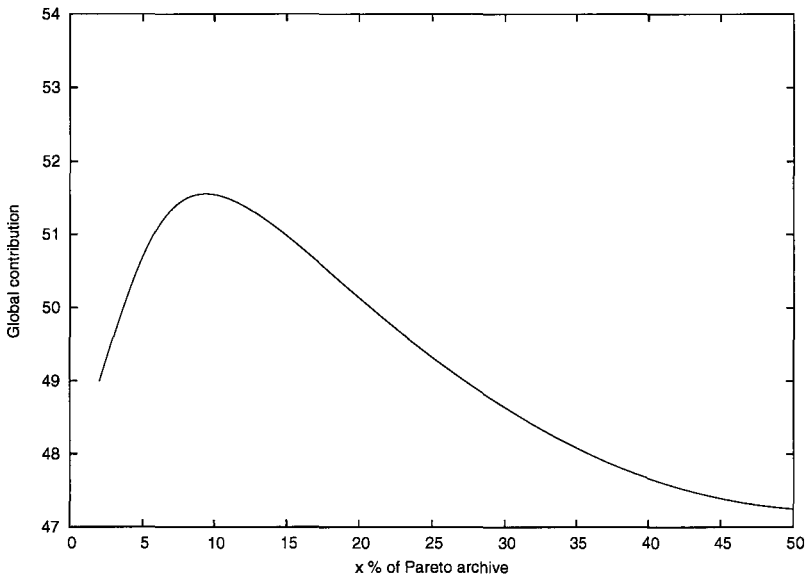



FIG. 6.7 – Contribution moyenne.

Le tableau 6.2 compare différentes fréquences de migration dans lesquels le nombre d'itérations séparant les migrations varie. Ainsi les migrations ont lieu toutes les 5, 10, 25, 50 ou 80 générations. Une fois encore, une configuration surpasse les autres. Il s'agit de la configuration toutes les 50 générations.

Ainsi, afin que la coopération soit performante il semble important d'échanger des solutions, mais pas trop ni trop souvent. Chaque île doit avoir le temps de converger.

TAB. 6.2 – Fréquence de migration (migration toutes les $x$ générations).

| | contribution moyenne | | | | |
|---|---|---|---|---|---|
| | 5 | 10 | 25 | 50 | 80 |
| 5 | - | 0.50 | 0.48 | 0.46 | 0.54 |
| 10 | 0.50 | - | 0.47 | 0.44 | 0.50 |
| 25 | 0.52 | 0.53 | - | 0.46 | 0.49 |
| 50 | **0.54** | **0.56** | **0.54** | - | **0.52** |
| 80 | 0.46 | 0.50 | 0.51 | 0.48 | - |

### 6.2.4 PARADISEO : plateforme de développement

Pour nos expérimentation sur le parallélisme, nous avons utilisé la plateforme ParadisEO de notre équipe [Cah05]. ParadisEO se définit basiquement comme une extension de la plate-forme EO. Elle adressent la conception de métaheuristiques, de nouveaux mécanismes et outils pour l'optimisation multi-objectif et principalement le déploiement de modèles parallèles et hybrides de métaheuristiques. A la mise en œuvre (Fig. 6.8), ces différents aspects font l'objet de modules indépendants et complémentaires respectivement désignés par « Moving Objects » (MO), « Multi-Objective Evolving Objects » (MOEO) et « Parallel and Distributed Evolving Objects » (ParaDisEO).

ParadisEO est un environnement de programmation pour les applications parallèles permettant la construction distribuée et à la volée du flot de données associé à l'exécution d'un programme. Elle exploite efficacement les architectures de type grappe avec différents niveaux de parallélisme (http ://www.lifl.fr/čahon/paradisEO/) et se définit comme une extension de la plate-forme EO. Il s'agit d'une plate-forme logicielle dédiée à la conception de métaheuristiques hybrides et/ou coopératives dans un environnement parallèle et/ou distribué.

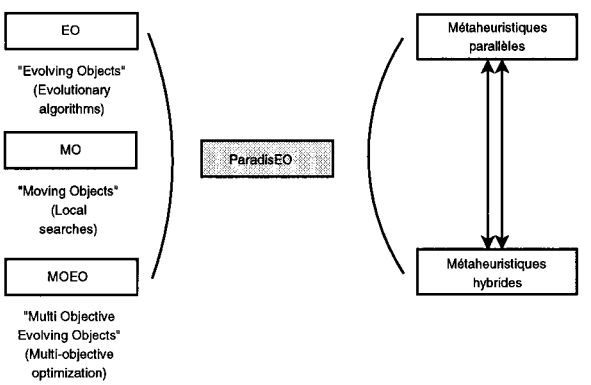



FIG. 6.8 – Organisation modulaire du ParadisEO

L'architecture du ParadisEO est multi-couche et modulaire (Fig. 6.9) offrant une haute flexibilité et adaptabilité, des hybridations de métaheuristiques plus simples à réaliser, et enfin une réutilisation maximale de code et de modèles. L'architecture repose sur trois couches identifiant trois groupes majeurs de classes : « *Solvers* » , « *Runners* » et « *Helpers* » .

- Les « *Helpers* » sont des classes de bas-niveau accomplissant des actions spécifiques en considération du processus d'évolution ou de recherche. On distingue deux catégories : « *Evolutionary Helpers* » (EH) et « *Local Search Helpers* » (LSH). Les EH regroupent essentiellement les opérateurs de sélection, de transformation, de remplacement, la fonction objectif et le critère de continuation.

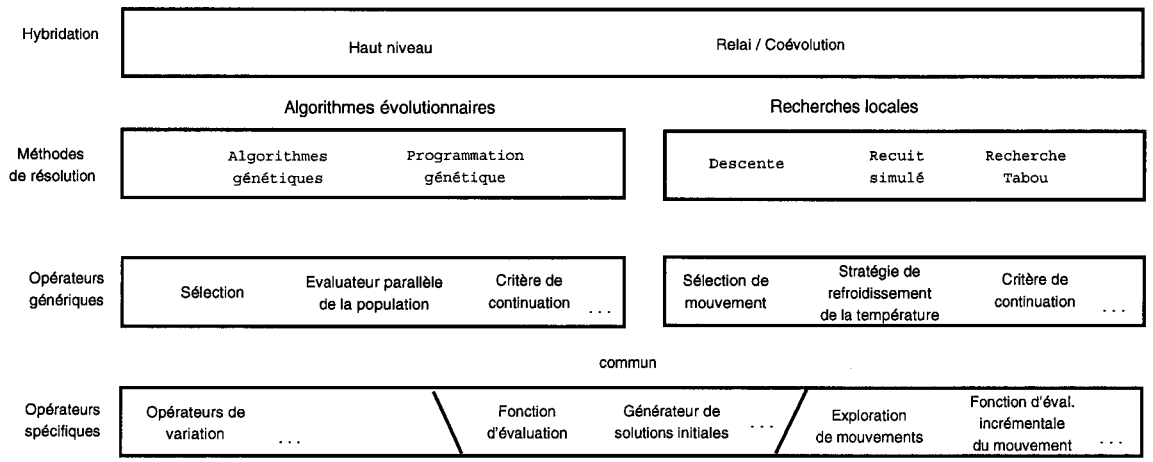



FIG. 6.9 – L'architecture de ParadisEO

- Les « *Runners* » caractérisent les classes implémentant les métaheuristiques elles-mêmes. Elles reposent sur l'exécution des métaheuristiques partant d'une solution ou d'un ensemble de solutions initiale(s) vers un état final. On distingue les « *Evolutionary Runners* » (ER) tels que les algorithmes génétiques, les stratégies évolutionistes, ...et les « *Local Search Runners* » (LSR) tels que la recherche Tabou ou le recuit simulé.

- Les « *Solvers* » sont voués au contrôle du processus d'évolution ou de recherche. Ils génèrent l'état initial (une solution ou population d'individus) et définissent la stratégie combinant et ordonnançant les différentes métaheuristiques. Les « *Solvers* » interagissent avec l'utilisateur par la saisie des données d'entrée et la délivrance de sorties (*e.g.* l'affichage de la meilleure solution, la production de statistiques, ...).

En considération du caractère « générique » des constituants intégrés, l'architecture se décompose en deux ensembles de classes : celles fournies « *Provided Classes* » (PC) et les autres non développées et donc requises « *Required Classes* » (RC). Les premières encapsulent la partie invariante des métaheuristiques. Elles sont donc génériques, implémentées dans la plate-forme, et garantissent le contrôle à l'exécution. Les secondes sont à concevoir par l'utilisateur. Elles intègrent les aspects spécifiques au problème ou à l'application. Ces classes sont identifiées dans ParadisEO. Le programmeur a la charge de les produire par la mise en œuvre du mécanisme de spécialisation.

#### 6.2.4.1 Le modèle en îles

Les composants gérant la coopération insulaire constituent des traitements à réaliser au terme de chaque génération, succédant à la phase de remplacement. On distinguera les formes synchrone ou asynchrone, parallèle ou distribuée. Nous présenterons ci-dessous pour illustration l'instantiation d'un tel opérateur sous une forme distribuée et asynchrone.

A la mise en œuvre, il convient de définir les différents paramètres qui vont caractériser

la régulation des migrations. Diverses stratégies sont ainsi envisageables quant au choix des solutions à émettre vers les populations voisines « eoSelect » , de la manière de les assimiler « eoReplacement » , de la topologie du modèle coopératif « eoTopology » et enfin du critère de décision d'immigration « eoContinue » . L'ensemble des composants nécessaires à la composition d'un gestionnaire de migrations asynchrone distribué sont définis à la figure 6.10.

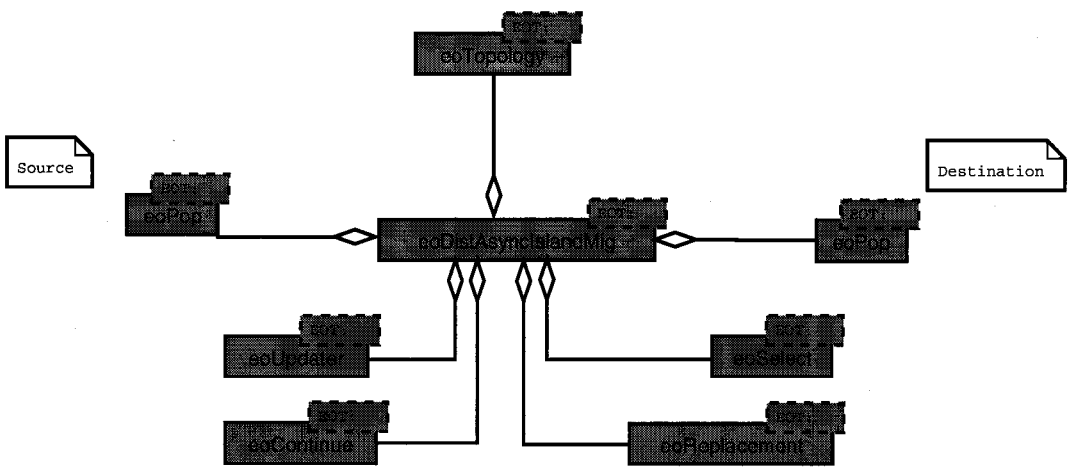



FIG. 6.10 – Une représentation UML des principaux composants nécessaires à la composition d'un gestionnaire de migrations asynchrone distribué.

Il est donc nécessaire de répondre à chacun des critères précédemment identifiés. Pour chacune des classes abstraites mentionnées (« eoContinue » , « eoSelect » , « eoReplacement » , « eoTopology » ), on dispose de nombreuses sous-classes les implémentant et correspondant à une stratégie donnée. Il n'y a pas de code supplémentaire à produire, puisque la mise en œuvre de ces mécanismes est indépendante du problème traité.

La spécification du gestionnaire de migrations synchrone est similaire. En lieu et place du décideur d'immigration, l'utilisateur paramétrera un nombre fini d'itérations entre deux phases d'échange. Une migration consiste alors à émettre des solutions vers les populations voisines, puis à attendre l'arrivée d'immigrants avant de poursuivre le processus d'évolution.

#### 6.2.4.2 La parallélisation de la fonction objectif

Ce modèle se base aussi sur le paradigme « Maître/Esclave » . Une solution est émise vers différents sites évaluateurs générant des valeurs de qualité partielles, ensuite collectées et aggrégées.

Les objets requis à l'instantiation (Fig. 6.11) sont les mêmes que ceux employés à la distribution de la phase d'évaluation de la population. Il convient de plus de fournir une fonction d'agrégation de qualités partielles. En effet, ce mécanisme s'avère dépendant de

l'application traitée.

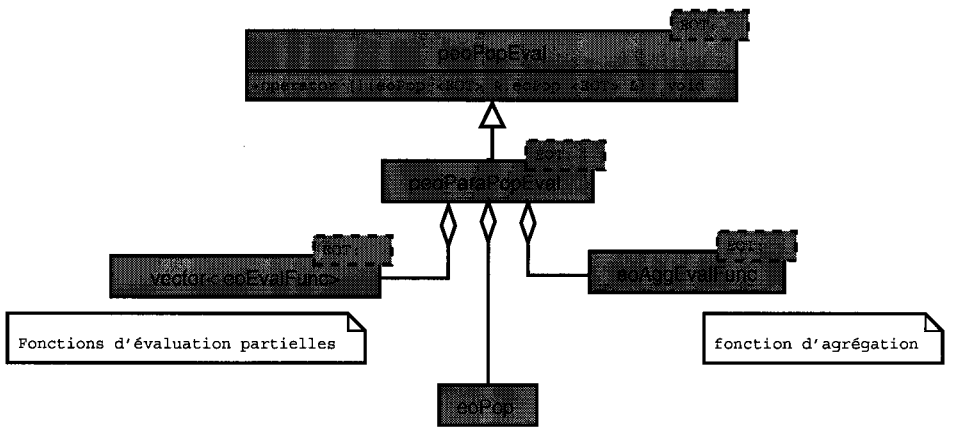



FIG. 6.11 – Principaux composants liés à la distribution de la fonction d'évaluation

#### 6.2.4.3 La parallélisation de la phase d'évaluation

Dans ce modèle, la population est centralisée sur une machine unique. Dans la forme parallèle, différents processus en exécution concurrente accèdent à une même population, partagée entre ces derniers. La population est partitionnée en sous-populations, qui seront séquentiellement émises vers différents noeuds évaluateurs, lesquels émettront en retour l'ensemble de valeurs de fitness calculées. Tant qu'il reste des solutions non évaluées, un processeur évaluateur en état inactif se verra attribuer une nouvelle tâche à accomplir. La taille des partitions de solutions constituées est un paramètre sensible aux performances à l'exécution. Elle doit être autant plus faible que le grain de calcul, *i.e.* le rapport entre le coût de l'évaluation d'une solution et de son temps de communication est important.

#### 6.2.4.4 Règles d'association avec ParadisEO

L'ensemble des composants instanciés et nécessaires à l'intégration d'un gestionnaire de migrations asynchrone au sein d'un A.E. sont définis dans la table 6.3. Il apparaît que la mise en oeuvre du modèle coopératif insulaire ne nécessite aucune connaissance supplémentaire du problème traité. A l'instanciation, tous les composants manipulés sont paramétrés par le type de solution manipulée « Rule» . La réutilisation est ici maximale.

Nous ferons quelques observations relatives au code énoncé ci-dessous. Afin de mettre en œuvre les phases de migration. Les composants instanciés, et implémentant respectivement les interfaces « eoContinue» , « eoSelect» ,« eoReplacement» et « eoTopology» , configurent le modèle en iles. Remarquons que deux instances de population sont fournies à l'instanciation du gestionnaire de migrations. Elles dénotent respectivement la population depuis laquelle seront constitués les solutions émigrantes, et celle dans laquelle seront intégrées les individus immigrants. Dans notre problème de recherche de règles d'association

| Opérateurs génériques | Opérateurs spécifiques |
|---|---|
| `eoFreqContinue`<br>(critère de décision de besoin d'immigration)<br>`eoRandomSelect`<br>(stratégie de sélection d'une solution à émigrer)<br>`eoSelectNumber`<br>(composition d'un ensemble de solutions émigrantes)<br>`eoReplacement`<br>(stratégie d'intégration des solutions immigrantes)<br>`eoRingTopology`<br>(topology d'interconnexion) | |

TAB. 6.3 – Identification des opérateurs, génériques ou spécifiques, nécessaires à l'instantiation d'un gestionnaire de migrations ascynchrones, sur environnement d'exécution distribué.

multi-critères, la population d'échange est l'ensemble des solutions Pareto (archive).

### 6.2.5 Validation du modèle parallèle

Les résultats sont obtenus à partir de plusieurs exécutions du programme sous plusieurs PC de type Pentium IV Intel 3 Ghz. Pour ces expérimentations, la base de données d'expression génique est utilisée (yeastDB). Les paramètres choisis sont :

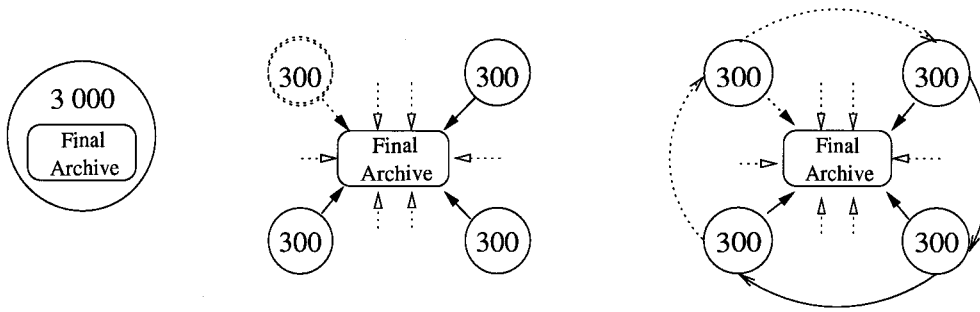
- taille de la population = 300,
- sélection = 2/ 3 (200),
- taux de mutation = 0.5,
- taux de croisement = 0.8,
- sélection dans l'archive Pareto (élitisme) = 0.5,
- nombre minimal de générations = 300,
- échange toutes les 50 générations de 10% de l'archive Pareto locale.

Afin de valider l'approche parallèle coopérative, trois configurations différentes ont été testées (voir figure 6.13). Ces configurations ont été choisies afin d'avoir une même population globale :

- *Conf 1 :* Un seul algorithme génétique, avec une population de **3 000** individus. L'archive Pareto de l'algorithme est l'archive finale.
- *Conf 2 :* Dix algorithmes génétiques indépendants d'une population de **300** individus chacun. Les dix algorithmes contribuent à l'archive Pareto finale.
- *Conf 3 :* Dix algorithmes génétiques coopérants d'une population de **300** individus chacun. Les dix algorithmes contribuent à l'archive Pareto finale.

```
  #define MIG_FREQ 10 /* Migration frequency */

  #define MIG_SIZE 10 /*Size of migrations */ #define REINIT_FREQ 5 /*

int main (int __argc, char * * __argv) {

  paradisEO :: init (__argc, __argv);

  eoArchive <Rule> archive; /* Archive */

  ..........................

  (programme de AG sous EO)

  ..........................


  /* Migration manager */
  eoFreqContinue <Rule> mig_cont (MIG_FREQ);


  eoRandomSelect <Rule> random_select; /* Random selection */
  eoSelectNumber <Rule> mig_select (random_select, MIG_SIZE);

  eoDominanceMap <RULE> dom_map ;
  eoParetoRanking <RULE> pareto_rank (dom_map) ;
  eoNDPlusReplacement<RULE> arch_replace(pareto_rank)

  eoRingTopology topo;
  eoDistSyncIslandMig <Rule> island_mig (mig_cont, mig_select, arch_replace, topo, archive, archive);
  checkpoint.add (island_mig);

  eoEasyEA <Rule> ea (checkpoint, eval, breed, replace); /* Evolutionary Algorithm */

  ea (pop);

  paradisEO :: finalize ();

  return 0;
}
```

FIG. 6.12 – Extrait du code.




a) Conf 1 b) Conf 2 c) Conf 3

FIG. 6.13 – Les trois configurations testées.

Le Tableau 6.4 compare en moyenne les fronts obtenus par 10 exécutions de chaque configuration.

TAB. 6.4 – Comparaison des configurations.

| | Contribution moyenne | | | Entropie moyenne | | |
|---|---|---|---|---|---|---|
| | $Conf1$ | $Conf2$ | $Conf3$ | $Conf1$ | $Conf2$ | $Conf3$ |
| $Conf1$ | - | 0.39 | 0.28 | - | 0.56 | 0.50 |
| $Conf2$ | **0.61** | - | 0.40 | **0.69** | - | 0.53 |
| $Conf3$ | **0.72** | **0.60** | - | **0.71** | **0.70** | - |

Le tableau 6.4 montre clairement que $Conf3 > Conf2 > Conf1$ que ce soit pour l'efficacité du front ou pour la diversité et atteste de l'intérêt de la coopération de méthodes semblables.

## 6.3 Approche hybride

Afin de tirer avantage des bénéfices apportés par différentes méthodes de résolutions, il est souvent nécessaire de les combiner. Aujourd'hui, les méthodes hybrides permettent d'obtenir les meilleurs résultats sur la plupart des problèmes, qu'ils soient académiques ou réels (voyageur de commerce, affectation quadratique, etc.). Nous nous sommes intéressé à l'hybridation entre métaheuristiques et méthodes exactes. Les méthodes exactes permettent de prouver l'optimalité des solutions pour des instances de taille raisonnable, alors que les méthodes approchées trouvent de bonnes solutions pour des instances de taille supérieure.

Voici une très brève présentation de la taxonomie sur la classification des métaheuristiques hybrides présentée dans [Tal02] :

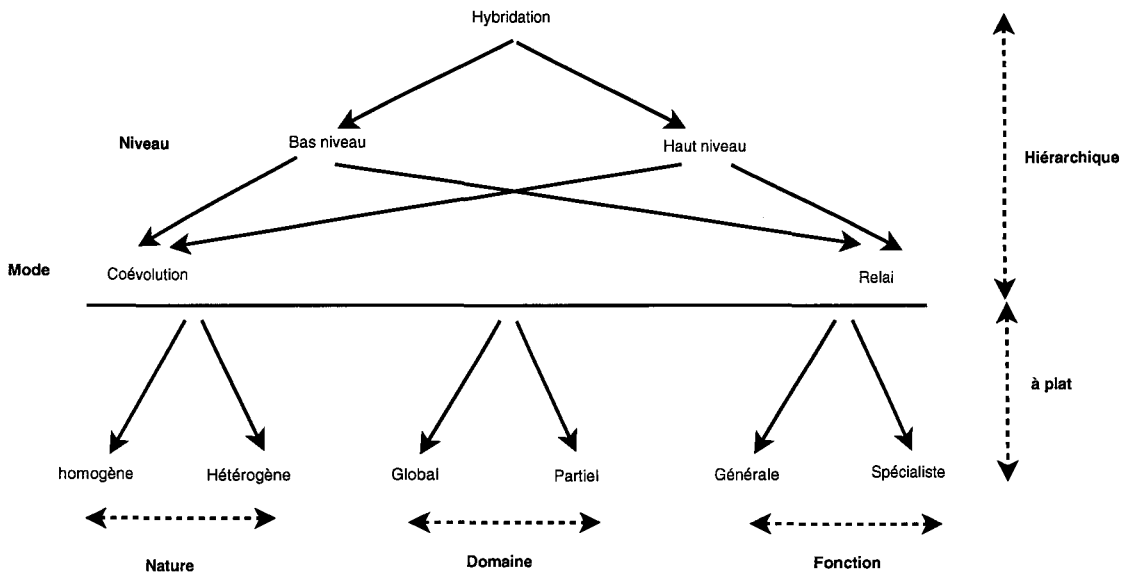



FIG. 6.14 – Classification desdes méthodes d'optimisation hybrides.

- Bas-niveau ou Haut-niveau : la première catégorie consiste en une composition de méthodes. Une fonction d'une métaheuristique est une autre métaheuristique. Pour le high-level, les différentes métaheuristiques sont imbriquées.
- Relay ou coévolution : les métaheuristiques sont appliquées l'une après l'aure. La sortie de l'une est l'entrée de la suivante. La coévolution représente des modèles de coopération, avec de la parallèlisation.
- Homogène et hétérogène : homogène signifie que tous les algorithmes utilisent la même métaheuristique, à l'opposé d'hétérogène.
- Global ou partiel : la recherche se fait sur l'ensemble de l'espace de recherche pour la caractérisation globale. A l'inverse, pour partiel, le problème est découpé en sous problèmes.
- Général ou spécialiste : une hybridation générale sera telle que tous ses algorithmes résoudront le même problème cible. En spécialiste, certains algorithmes peuvent travailler sur le problème transformé (reformulation en un autre problème d'optimisation)

### 6.3.1 Méthode exacte (Procédure énumérative)

Il n'existe pas de méthode exacte (énumérative) pour la recherche de règles d'association multi-objectif. La méthode la plus utilisée pour la recherche de règles d'association est l'algorithme Apriori qui est basé sur un principe d'énumération. Cet algorithme réalise une énumération efficace de toutes les règles verifiant un *support* minimal et une *confiance* minimale. L'efficacité de l'algorithme se base sur la propriété de monotonie du *support* qui permet de rechercher les ensembles fréquents de taille $k$ seulement à partir des ensemble fréquents de taille $k-1$. Cependant cet algorithme n'est pas transposable à d'autres critères que le *support*, car les autres critères permettant de mesurer la qualité des règles ne sont pas monotones, ce qui a pour conséquence que la valeur du critère pour un ensemble

peut être meilleure que pour l'un de ses sous-ensembles, ce qui n'est pas le cas du *support*.

Nous proposons alors une méthode énumérative permettant l'utilisation de cinq critères.

Les caractéristiques de cette méthode sont les suivantes :
- En entrée : l'ensemble des attributs,
- Enumération pour toutes les configurations possibles de toutes les valeurs possibles,
- Evaluation selon les cinq critères,
- En sortie : un ensemble des meilleures règles (dans une archive Pareto locale).

### Énumération des différentes combinaisons de valeurs

Soit $n$ le nombre d'attributs total.
Les $n-1$ attributs ont une valeur d'attribuée (l'initialisation se fera avec la première valeur de chacun). Les valeurs attribuées sont grisées sur la figure 6.15. Les règles sont alors générées en énumérant les valeurs que peut prendre le dernier attribut.
Puis une valeur des n-1 attributs est changée. Ainsi, des règles différentes seront générées en énumérant à nouveau les valeurs que peut prendre le dernier attribut.

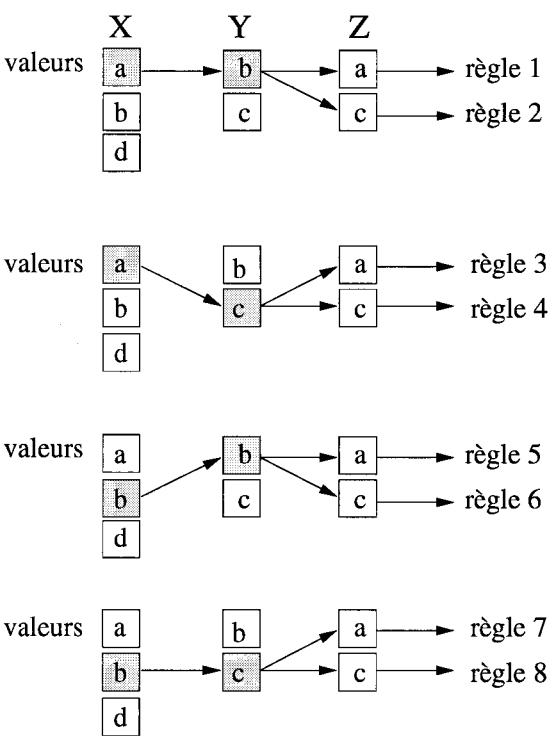



FIG. 6.15 – Énumération des différentes combinaisons de valeurs.

### Énumération des combinaisons d'attributs de la règle :

Soit $n$ le nombre d'attributs total.

Le principe est identique au précédent, appliqué aux attributs, et itéré pour toute taille de règle (voir figure 6.16). Sur l'exemple, tous les ensembles de trois conditions sont cherchés parmi quatres attributs **A**, **B**, **C**, **D** :

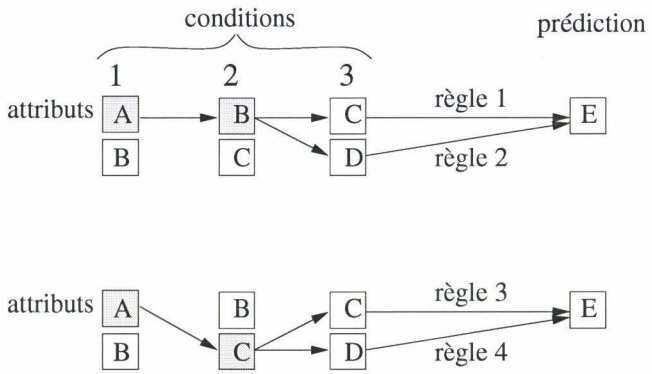



FIG. 6.16 – Énumération de toutes les combinaisons d'attributs de la règle.

Le nombre de solutions qu'une énumération complète devrait générer est tel que :
Soit $n$ le nombre d'attributs total, $v(att_i)$ le nombre de valeurs de l'attribut i, $l$ la longueur d'une règle quelconque, et $tailleMax$ la taille maximum désirée pour les règles produites.

- Le nombre de règles différentes de taille $l$. Pour $n$ prédictions, il doit y avoir $l-1$ conditions :

$$n \cdot C_{n-1}^{l-1}$$

- Le nombre de combinaisons de valeur pour une règle de taille $l$ :

$$\prod_{i=0}^{l} v(att_i)$$

Le nombre de règles issues d'une énumération exhaustive est alors :

$$nb\_regle = n \cdot \sum_{l=2}^{tailleMax} C_{n-1}^{l-1} \cdot \prod_{i=0}^{l} v(att_i)$$

Par exemple, pour 10 attributs ayant chacun 5 valeurs possibles, il y a 24 242 250 règles de taille strictement inférieure à 7. Il est donc impensable de générer des règles pour un grand nombre d'attributs.

### 6.3.2 Schéma d'hybridation AG/exacte

La procédure énumérative est utilisée comme un opérateur de croisement lorsque le nombre d'attributs différents composant les deux règles parentes n'est pas trop élevé (voir

figure 6.17). Ainsi, la procédure énumérative va explorer la région déterminée par les attributs participants aux règles parentes. Le résultat d'un tel opérateur consiste en une archive Pareto locale qui est ensuite utilisée pour mettre à jour l'archive globale et pour générer les enfants. La procédure d'hybridation est décrite ci-dessous :

---

`Crossover(`$Rule_1, Rule_2$`)`

---

```
{
```

$AttributeSet \leftarrow Attrib_{Rule_1} \cup Attrib_{Rule_2}$ `// Construction de l'ensemble des attributs`

$nb \leftarrow \mid AttributeSet \mid$ `// calcul du nombre d'attributs`

```
if (nb ≤ MaxNb)
    EnumProc(AttributeSet,nb)
else
   NormalCrossover(Rule1, Rule2)
}
```

---

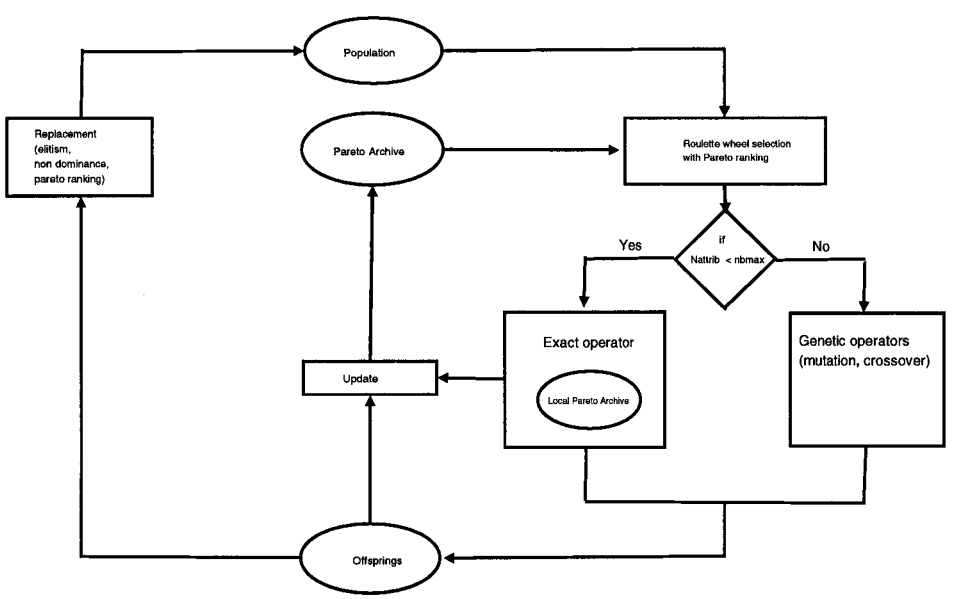



FIG. 6.17 – Hybridation : algorithme génétique multi-objectif et procédure énumérative.

### 6.3.3 Evaluation du modèle

Pour évaluer le modèle hybride, des expérimentations sur le même jeu de données d'expression génique qu'au chapitre précédent ont été réalisées. Mais dans ce cas un front

TAB. 6.5 – Performances des différentes versions.

| Fréquence d'application | Contribution moyenne | D-metric moyenne |
|---|---|---|
| 0% | 0.03 | 5.20 |
| 20% | 0.18 | 4.47 |
| 50% | 0.22 | 4.04 |
| 100% | 0.29 | 3.80 |

de référence a été utilisé. Ce front a été construit en prenant l'intersection de tous les fronts trouvés au cours des différentes expérimentations que nous avons menées.

Le tableau 6.5 montre la contribution de l'utilisation de l'opérateur exact. Les différentes configurations font varier la fréquence de l'utilisation de cet opérateur depuis 0% (pas d'appel à l'opérateur) jusqu'à 100% (l'opérateur est utilisé toutes les générations). Dans la première colonne, la mesure de contribution est utilisée et une moyenne sur 10 exécutions est indiquée. Il est clair que la version à 100% surpasse les autres versions. Pourtant, comme le montre la figure 6.18, l'accroissement n'est pas linéaire et même une fréquence moyenne permet une nette amélioration de la qualité des fronts obtenus.

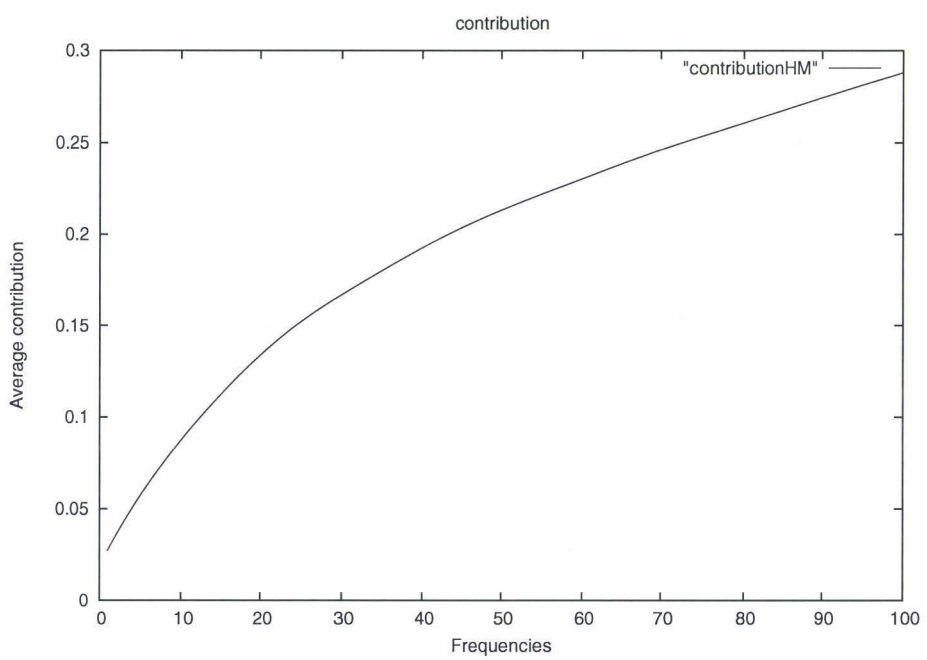



FIG. 6.18 – Evolution de la contribution moyenne en fonction de la fréquence d'utilisation de l'opérateur exact.

La deuxième colonne compare les fronts obtenus avec les différentes fréquences à l'aide

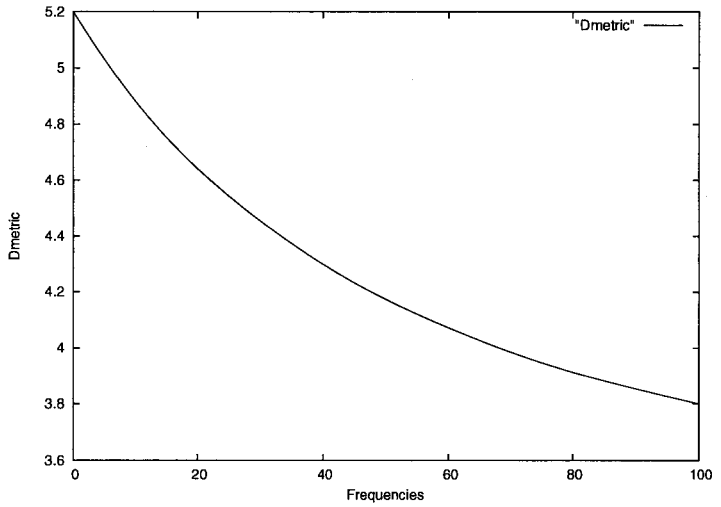



FIG. 6.19 – Evolution de la D-metric en fonction de la fréquence d'utilisation de l'opérateur exact.

de la D-métrique. Cette métrique compare deux fronts en utilisant la S-métrique qui mesure l'hyper volume de l'espace objectif dominé par un front. La D-métrique compare la S-métrique obtenue pour chacun des fronts. Plus le chiffre donné est petit, plus l'hyper volume dominé par le front de référence est petit et donc le front comparé est meilleur. Ces résultats confirment bien évidemment la suprématie de la version à 100% avec la même remarque sur la non linéarité de cette augmentation qui est visualisée sur la figure 6.19.

## 6.4 Applications : intégration du module

Afin de valider la pertinence de notre approche et les performances de l'algorithme proposé, deux intégrations ont été réalisées : l'une dans le cadre du projet GGM (Grille GenoMédicale) de l'ACI masse de données, l'autre dans la plateforme bioinformatique BASE.

### 6.4.1 Projet GGM : vers une grille biomédicale

Notre algorithme génétique multi-objectif pour les règles d'association à été adapté et intégré dans le service de datamining pour le projet GGM.

Le projet GGM est un projet lancé en juillet 2004 et financé par l'ACI Masse de Données avec la collaboration de trois laboratoires de recherche (LIRIS à Lyon, LIFL à Lille, IRIT à Toulouses).

Ce projet vise à proposer une architecture logicielle, s'appuyant sur les grilles de calcul, capable de gérer des données hétérogènes et dynamiques au sein d'entrepôts de données

distribués, à des fins d'analyse et de traitement intensifs.

Ce challenge est particulièrement important dans le cadre des grilles biomédicales. En effet, la diffusion des technologies haut débit en génomique/protéomique et la gestion informatique du dossier médical réparti ouvrent des perspectives diagnostiques totalement novatrices. Parce qu'elles exigent une capacité d'analyse et de traitement considérable et un partage d'informations hétérogènes et très volumineuses à grande échelle, ces technologies apparaissent comme des "cibles" naturelles des grilles de calcul.

Trois services principaux été identifiés :

- Service d'entrepôts de données : Gestion de données (hétérogénéité, dynamicité, sécurité, traçabilité, efficacité d'accès).
- Service de datamining : porter sur grille de calcul des algorithmes d'extraction de connaissances (datamining) sur des masses importantes de données dynamiques et hétérogènes réparties à grande échelle.
- Service de requêtes : réaliser des mandataires sémantiques pour optimiser/réguler l'utilisation des ressources (calcul, stockage) et assurer une adaptation des données aux droits et besoins des utilisateurs finaux doivent être développés.

Les équipes partenaires du projet (LIRIS, IRIT, LIFL), très complémentaires, sont fortement impliquées dans des initiatives en grid computing, fouille de données, algorithmique parallèle, systèmes d'information et entrepôts médicaux.

Ainsi, le LIRIS apporte une contribution au niveau des entrepôts de données, et des mandataires collaboratifs. L'IRIT participe au niveau du service de requêtes. Notre contribution (LIFL) se situe au niveau du service de datamining. L'accent est mis sur les modèles de data mining basés sur les règles d'association.

Le schéma de la figure 6.20 représente l'architecture globale du projet :

Mandataire pour l'accès aux données : cette tâche est constituée de deux sous-tâche, la première pour de l'accès basic ou sécurisé aux données des hôpitaux et des centres de recherche en génétique, et la seconde pour la gestion de caches collaboratifs.

Service de requêtes : cette tâche s'appuie sur les mandataires d'accès au données pour délivrer efficacement les données au service d'entrepôt de données.

Service d'entrepôt de données : cette tâche est divisé en deux sous-tâches, l'une pour la construction de l'entrepôt, l'autre pour pour l'utilisation de l'entrepôt. Cette tache s'appuiera sur le service de requêtes et le service de mandataire.

### 6.4.2 BASE et le plugin Rule mining

Afin de rendre notre algorithme publique et disponible à la communauté bio-informatique, notre algorithme génétique multi-objectif parallèle hybride pour les règles d'association a été adapté pour la plateforme BASE sous forme d'un plugin appelé Rulemining.

En effet, BASE possède également des plugins de normalisation, de visualisation et d'analyse des données. Peu de développements de plugins dans BASE portent sur la fouille de données (datamining).

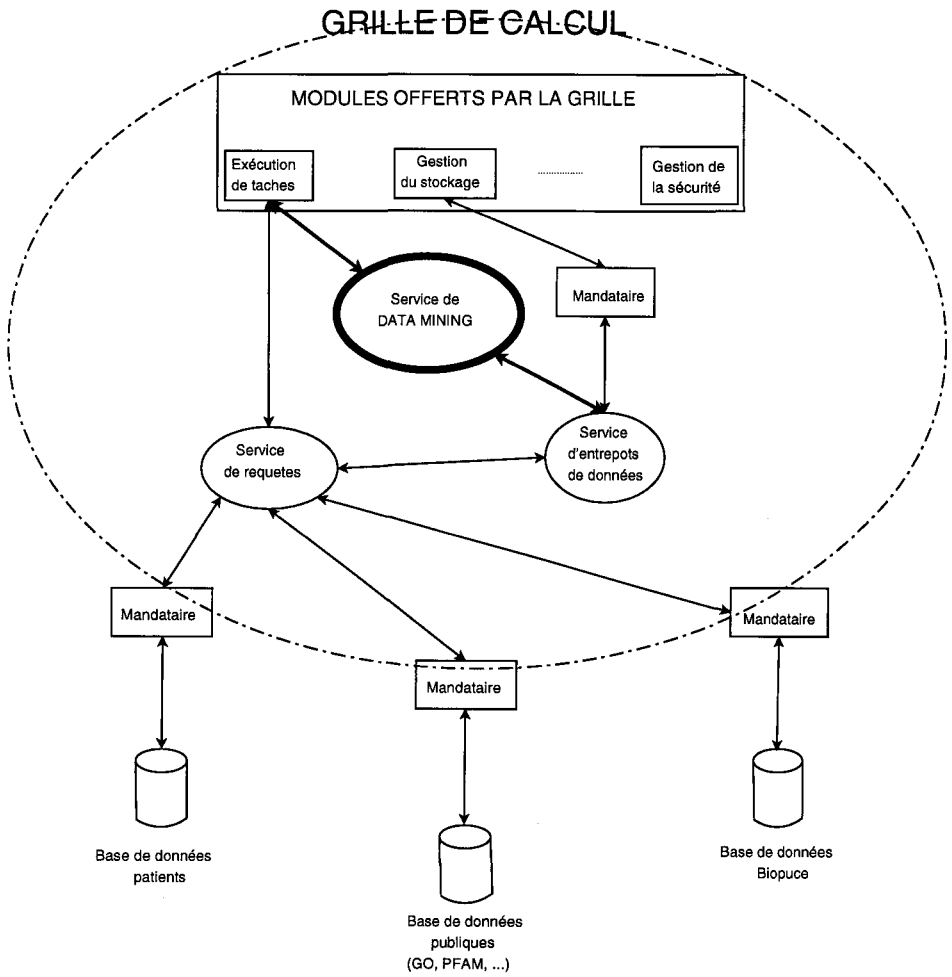



FIG. 6.20 – Architecture logicielle du projet GGM

BASE [3] est une base relationnelle très complète de l'équipe de C. Peterson à Lund University en Suède [STVC+02] qui bénéficie d'un développement très actif. Des mises à jour avec de nouvelles fonctionnalités sont publiées plusieurs fois par an. De plus, le logiciel dispose d'un système de modules (plug-in) que les développeurs peuvent ajouter indépendamment pour l'adapter à différents besoins. Ainsi, des contributeurs externes peuvent proposer d'intégrer de nouveaux systèmes de traitement de données qui n'existent pas dans la base originelle. BASE permet de stocker les fichiers de résultats issus de différents types de plates-formes et de les lier à divers paramètres expérimentaux, de protocoles et de la fabrication des lames. Elle dispose d'un système de requête et de filtrage efficace, de plusieurs options de visualisation des données de modules permettant d'effectuer différents types de normalisation, d'analyses statistiques et de datamining. Les données peuvent être extraites sous divers formats compatibles avec les logiciels d'analyses les plus utilisés.

BASE respecte le standard MIAME (Minimum Information About Microarray Experiment) [BHQ01] qui a pour but de définir le minimum d'informations requises pour interpréter de façon non ambiguë les données de puces à ADN (ensemble de renseignements à fournir lors de toute publication). Toutes les étapes, de la fabrication des lames et de l'extraction des ARN à l'analyse des données, sont ainsi détaillées dans la base de données de BASE.

### 6.4.3 Schéma de BASE

Initialement conçu en PHP et C, récemment avec la la sortie de BASE v. 2.0 entièrement rédigée en Java. BASE est plus qu'un logiciel de stockage et de tri des informations, il intègre également des applets de visualisation de données et des modules d'analyse performants avec la prise en charge du format MAGE-ML qui permet d'exporter les données structurées selon la norme internationale MIAME [BHQ01].

L'interface de BASE se décompse en six grandes parties présentées dans le bandeau gauche de la page principale (figure 6.21) :

- **REPORTERS** : Dans BASE, les sondes (gènes) qui sont déposées sur les lames sont appelées *Reporters.* On trouve dans cette partie des informations telles que la position sur le génome ou le nom du gène que la sonde représente. Chaque sonde doit avoir un "reporter ID" (identifiant unique). Les reporters sont créés automatiquement à l'ajout de puces (voir " ARRAY LIMS ").
- **ARRAY LIMS (Laboratory Information Management System)** : gère toutes les informations sur la disposition du matériel biologique dans les puces, l'annotation des puits ainsi que sur la production des puces.
- **BIOMATERIALS** : Cette partie gère les données qui sont l'origine de l'hybridation. En effet, elle permet de définir et visualiser les échantillons biologiques (cibles), ainsi que les étapes d'extraction et de marquage. Des informations comme les quantités de matériel biologique et les différents protocoles utilisés peuvent être sauvegardées.

[3] http ://base.thep.lu.se

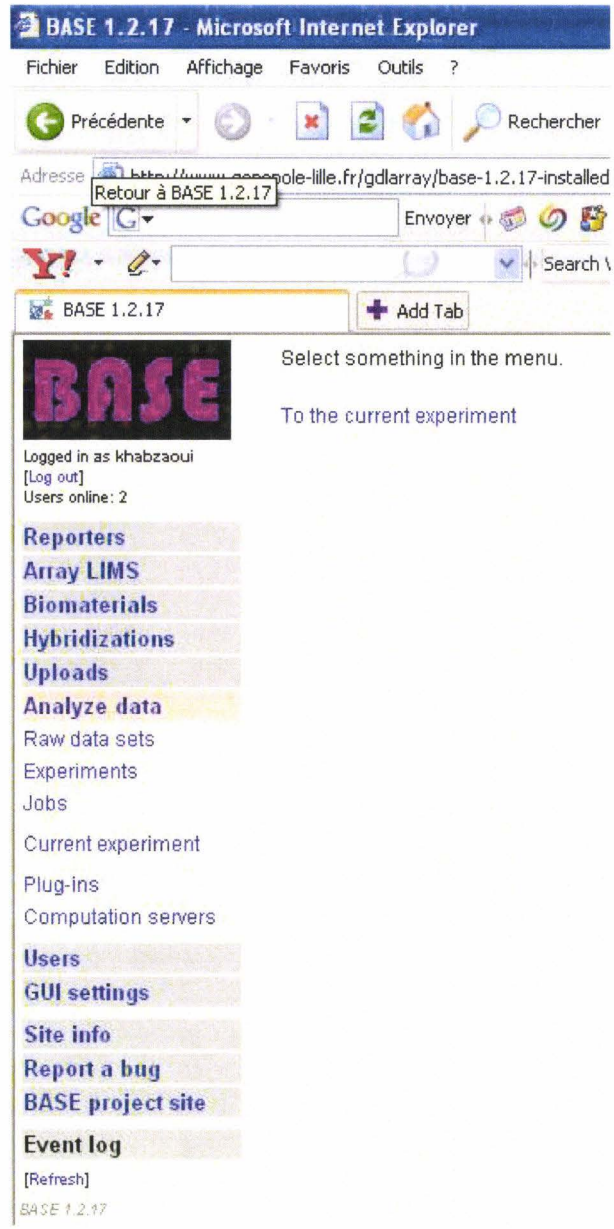



FIG. 6.21 – Grandes parties de l'interface principale de BASE.

- **HYBRIDIZATIONS** : La partie HYBRIDIZATION est le noeud central de BASE puisqu'elle relie les matériels biologiques avec les puces utilisées, et avec l'analyse des données.
- **UPLOADS** : Cette partie est utilisée pour charger les fichiers dans BASE. Une fois chargés dans BASE, ces fichiers sont accessibles à partir de toutes les autres sections de BASE. Cette section est désignée comme une section temporaire de stockage pour les fichiers qui ne sont pas encore associés ou utilisés à leur fin.
- **ANALYZE DATA** : Cette partie permet l'analyse complète de jeux de données associés à une expérience bien définie. Dans cette partie, la manipulation des donnée de plusieurs puces à analyser, s'efffectue à partir de modules (plugin). Ces derniers peuvent être ajoutés indépendamment par des dévelopeurs, pour adapter la base aux différents besoins d'analyses.
  La rubrique **Plug-ins** contient la liste de toutes les applications (plug-ins) disponibles dans BASE par exemple Hierarchical clustering (la classification hiérarchique), MDS (la graduation multidimensionnelle est une manière non linéaire de réduire la dimensionnalité d'un jeu de données), PCA (analyse en composante principale), Normalisation par rapport à la médiane, Normalisation Lowess ainsi que notre boite de règle d'association en datamining (rule mining).

#### 6.4.3.1 Le plugin Rule mining

L'algorithme génétique multi-objectif parallèle utilisant le framework Evoluting Object ( EO ) et ParadisEO pour les règles d'association a été adapté pour la plateforme BASE sous le plugin Rulemining (voir figure 6.22) afin d'extraire des règles d'associations entre gènes. Cette méthode permet de mettre en évidence des relations plus précises entre les gènes.

Nous avons intégré aussi un outil développé au sein de notre équipe permettant la visualisation des règles d'association générées par l'algorithme génétique (voir la figure 6.23). Cet outil permet de générer deux types de vue différentes à partir des règles sélectionnées : une représentation en trois dimensions et une en deux dimensions.

**6.4.3.1.1 Visualisation 3D** Nous avons basé notre représentation 3D sur celle présentée dans [WWT99] qui peut visualiser beaucoup de règles d'association. Les lignes de la matrice deux dimensions servant de support à la représentation représentent les attributs et les colonnes représentent les associations d'attributs. Les blocs verts (resp. rouges) de chaque colonne (règle) représentent la condition (ou l'antécédent) et la prévision (ou la conséquence). Les identités des attributs sont indiquées le long de la matrice.
Pour pouvoir évaluer chaque règle pas seulement grâce au support et à la confiance, nous ajoutons à la représentation 3D la possiblité de voir les différentes mesures de qualité calculées par notre algorithme de génération. On peut sur la visualisation 3D, grâce à la souris, réaliser des zoom, rotation et translation. La figure 6.24 montre une visualisation en trois dimensions des résultats obtenus sur la base de données biopuces *DB1*.

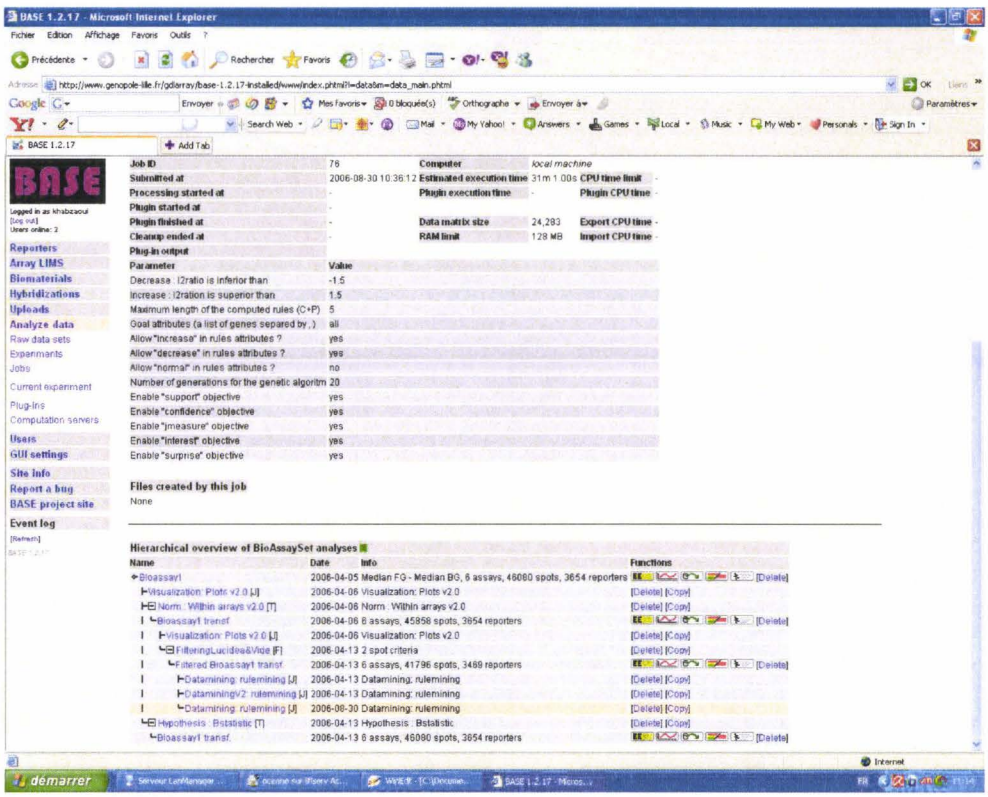



FIG. 6.22 – Plugin Rule mining.

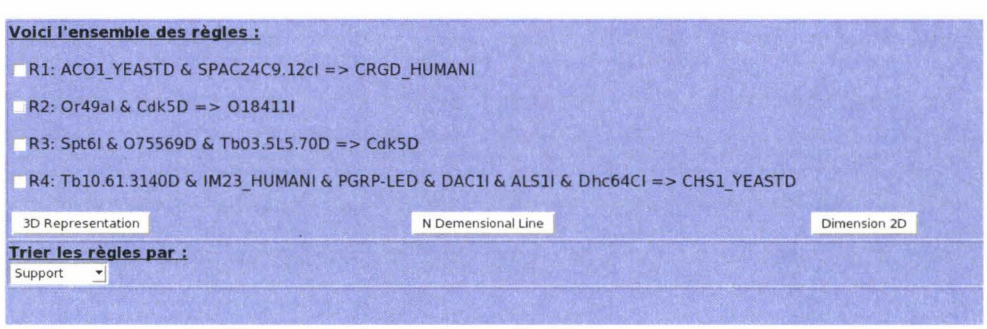



FIG. 6.23 – Règles d'association

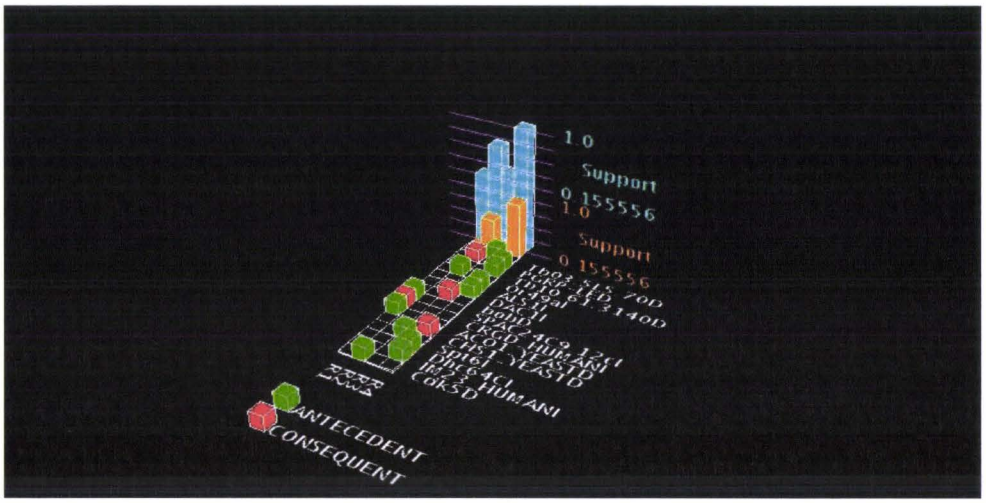



FIG. 6.24 – Visualisation en 3 Dimentions des résultats obtenus par l'algorithme génétique pour recherche de règles d'association (format de règles).

**6.4.3.1.2 Visualisation de tous les critères** La deuxième visualisation est héritée de la visualisation N Dimensional Line [ID90] qui permet de comparer un grand nombre de règles par rapport à un grand nombre de critères. Chaque ligne représente une règle et chaque critère est un point de l'axe des abscisses. La valeur des critères est tracée sur l'axe des ordonnées. Nous avons pu remarquer qu'un des inconvénients de cette visualisation vient de la perte de lisibilité lorsque plusieurs règles ont des valeurs proches et qu'elles se superposent. Pour remédier à ce problème, nous avons réalisé une normalisation de l'échelle afin de la rendre indépendante pour chaque critère. La figure 6.25 montre une visualisation de tous les critères des résultats obtenus sur la base de données biopuce *DB1*.

**6.4.3.1.3 Enrichissement des règles avec GO (Gene Ontology)** L'objectif de l'ontologie GO est d'établir un vocabulaire structuré, contrôlé et dynamique pour décrire le rôle des gènes et produits de gènes. Elle permet entre autre, d'interpréter les résultats d'analyse des données issues des expérimentations de puces à ADN. Aussi, pour profiter de cette connaissance, nous avons mis en place un outil qui permet de mettre en relation les résultats de nos algorithmes génétiques recherchant des règles d'association entre gènes, avec les groupes GO des gènes présents dans les règles.

Ainsi, cet outil considère l'ensemble des règles résultat et analyse pour chacune de ces règles les groupes GO participant à la règle. Les groupes GO étant organisés sous forme d'un graphe orienté acyclique (voir chapitre 2), l'outil d'analyse va rechercher pour chaque règle, les groupes GO communs aux différents gènes afin de mettre en évidence le groupe GO commun le plus précis (père des différents gènes).

L'outil permet donc d'afficher le résultat de l'analyse avec GO et de proposer de télécharger le fichier résultat (qui contient les règles et les conclusions d'analyse avec GO). Dans la figure 6.26, la catégorie "Meilleurs" indique les groupes GO communs des gènes impliqués dans la règle d'association.

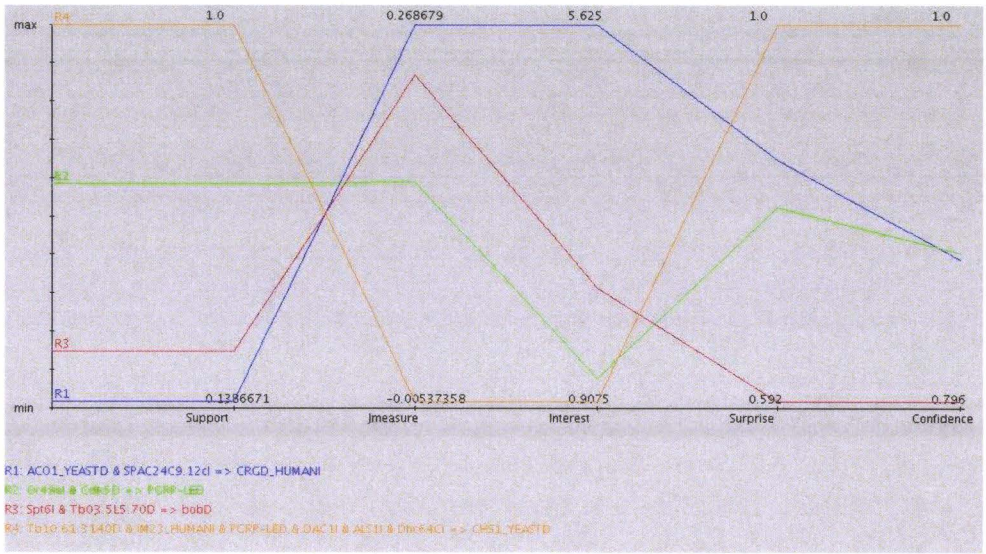



FIG. 6.25 – Visualisation en 2 Dimentions des résultats obtenus par l'algorithme génétique pour recherche de règles d'association.

```
Regle: R1
        Condition:
                ACO1_YEAST
                bill
                SPAC24C9.12c
        Conclusion:
                CRGD_HUMAN
        Genes non renseignes:
                bill
        Groupes GO communs:
                http://www.geneontology.org/go#GO:0007582
                        Groupe Parent: http://www.geneontology.org/go#GO:0008150
                http://www.geneontology.org/go#GO:0008150
                        Groupe Parent: http://www.geneontology.org/go#all
                http://www.geneontology.org/go#all
                http://www.geneontology.org/go#GO:0003674
                        Groupe Parent: http://www.geneontology.org/go#all
        Meilleurs:
                http://www.geneontology.org/go#GO:0007582
                http://www.geneontology.org/go#GO:0003674

Regle: R2
        Condition:
                Or49a
                Cdk5
        Conclusion:
                O18411
        Groupes GO communs:
                http://www.geneontology.org/go#GO:0003674
                        Groupe Parent: http://www.geneontology.org/go#all
                http://www.geneontology.org/go#all
                http://www.geneontology.org/go#GO:0007582
                        Groupe Parent: http://www.geneontology.org/go#GO:0008150
                http://www.geneontology.org/go#GO:0008150
                        Groupe Parent: http://www.geneontology.org/go#all
        Meilleurs:
                http://www.geneontology.org/go#GO:0003674
                http://www.geneontology.org/go#GO:0007582
```

FIG. 6.26 – GO et règle d'association.

## 6.5 Conclusion

Dans ce chapitre, nous avons présenté un algorithme génétique parallèle et une méthode énumérative pour traiter le problème de recherche de règles d'association multi-objectifs.

Les expérimentations permettent de confirmer l'utilité de la coopération avec une procédure énumérative. Ces expérimentations montrent aussi qu'il n'est pas forcément nécessaire d'utiliser la procédure d'énumération à chaque itération si le temps d'exécution est un critère important. En effet, puisque la procédure énumérative est la partie la plus coûteuse, l'utilisation de cette procédure deux fois plus souvent conduira à un temps d'exécution deux fois plus long. Ainsi, dans un soucis de compromis entre temps d'exécution de la recherche et qualité des solutions obtenues, il peut être intéressant de n'exécuter la procédure qu'une fois sur deux, par exemple. Ceci conduit, dans le cas des données étudiées, à une diminution du temps de 50% pour une qualité diminuée d'environ 25%.

L'approche coopérative semble donc très prometteuse. Nous avons montré à la fois l'apport du parallélisme et l'apport de coopération entre méta/exacte.

# Chapitre 7

# Conclusion générale et perspectives

Dans cette thèse, nous avons abordé et étudié le problème de recherche de règles d'association en extraction de connaissances dans le domaine des puces à ADN avec une approche évolutionnaire multi-objectif. Ainsi, plusieurs contributions ont été apportées. D'abord, nous avons modélisé le problème de la recherche de règles d'association comme un problème d'optimisation multi-objectif et avons ensuite développé plusieurs versions d'un algorithme de résolution évolutionnaire (séquentiel, parallèle et hybride). Ces algorithmes ont été intégrés dans des applications afin de les rendre disponible.

Dans ce mémoire, nous avons donc dans un premier temps présenté les puces à ADN et ses technologies ainsi que les techniques d'analyse de données. En raison du nombre important de gènes et de la complexité des réseaux géniques, les techniques de dataming, d'apprentissage et de statistique se sont avérées un outil très utile pour l'analyse de profils d'expression. Parmi les techniques utilisées, nous citons la classification, le clustering. Ces techniques ont été largement utilisées pour identifier des groupes de gènes partageant des profils d'expression similaires et les résultats obtenus sont très concluants. Néanmoins, ces méthodes ne permettent de découvrir qu'une partie des relations parmi toutes les relations potentielles entre les gènes. Ces méthodes ne donnent pas la relation exacte qui peut exister entre deux gènes ou deux groupes, et permette de donner qu'une image globale, l'information à un niveau plus local pouvant alors être perdue. C'est pourquoi nous avons proposé d'utiliser les règles d'association.

Nous avons présenté dans le deuxième chapitre les méthodes classiques (la famille des algorithmes Apriori) de résolution pour le problème des règles d'association. Ces méthodes ont montré leur efficacité pour trouver des solutions satisfaisantes pour un grand nombre de problèmes de petite taille et pour deux critères (le support et la confiance). Cependant, ces méthodes ne sont pas adaptables pour une résolution multi-objectif. Le plus gros problème concerne l'impossibilité de mesurer la qualité d'une solution sans utiliser le support et la confiance et d'introduire les mécanismes multi-objectif.

Nous avons ensuite exposé en premier lieu différents indicateurs mesurant la qualité des règles d'association (les plus utilisés). Puis, nous avons évoqué les propriétés et préférences

proposées par différents auteurs : Tan et al. [TKS02], Hilderman [HH99], V. Shi et al. [SDP01] et Piatetsky-Shapiro [PS91]. Ces travaux apportant un éclairage probabiliste et théorique pour caractériser une bonne mesure de qualité d'une règle d'association.

Puis, nous avons proposé une approche statistique. Le but de cette étude est de mettre en évidence les corrélations existantes entre les mesures afin de regrouper les critères ayant le même comportement. Nous avons donc déterminé un ensemble cohérent de 5 critères complémentaires. Nous nous plaçons dans le cadre de problèmes d'optimisation où il n'existe pas de modèle de préférences sur les critères en utilisant cinq objectifs (un représentant par groupe pour les cinq premiers groupes). Ces critères choisis sont : le *support*, la *confiance*, la *J-mesure*, l'*intérêt* et la *surprise*.

Dans le chapitre 4, nous avons présenté les principaux concepts de l'optimisation multi-objectif et les différentes approches de résolution pour traiter un problème d'optimisation mono ou multi-objectif : les méthodes exactes et les méthodes approchées. Ainsi, nous avons présenté brièvement les métaheuristiques les plus connues pour l'optimisation mono et multi-objectif. Récemment, beaucoup de recherches ont été menées sur l'application des algorithmes évolutionnaires aux problèmes d'optimisation multi-objectif. Celles-ci ont permis de mettre en avant l'intérêt d'utiliser des méthodes d'optimisation basées sur le concept de population. Dans la fin de ce chapitre, nous avons présenté un état de l'art sur les différentes mesures d'évaluation de la qualité d'un front Pareto. En effet, l'évaluation des résultats d'un algorithme multi-objectif est un problème délicat qui oblige à utiliser plusieurs mesures différentes, car il est impossible de représenter par une seule valeur la qualité des solutions, la taille du front et la répartition des solutions sur le front.

Le problème de la recherche de règles d'association peut donc se définir comme un problème d'optimisation combinatoire multi-objectif. Nous avons proposé un algorithme génétique permettant de traiter des bases de données classiquement utilisées en data mining (*UCI Machine Learning Repository*), ainsi que des bases de données relatives à des expérimentations sur puces à ADN. Cet algorithme possède un codage et des opérateurs adaptés à la recherche de règles d'association et des mécanismes multi-objectif ont été implémentés. Nous avons mis en place un mécanisme adaptatif pour pouvoir appliquer plusieurs mutations selon l'évolution de l'algorithme et adapter leur taux d'application en fonction de l'amélioration apportée par chacun d'eux.

Dans le dernier chapitre, nous avons proposé plusieurs approches coopératives. Nous avons montré à la fois l'apport du parallélisme et l'apport de la coopération entre méthodes de différents types.

Ainsi, nous avons présenté dans un premier temps une approche parallèle développée pour le problème de recherche de règles, dans laquelle différents algorithmes génétiques coopèrent. Puis dans un deuxième temps nous avons présenté une approche hybride entre une métaheuristique (algorithme génétique) et une méthode exacte (un algorithme énumératif). Il n'existe pas de méthode énumérative pour la recherche de règles d'association multicritère. La méthode exacte la plus utilisée pour la recherche de règles d'association est l'algorithme Apriori qui est basé sur un principe d'énumération. Cet algorithme réalise une énumération efficace de toutes les règles verifiant un *support* minimal et une *confiance* minimale. L'efficacité de l'algorithme se base sur la propriété de monotonie du *support*.

Cependant cet algorithme n'est pas transposable à d'autres critères que le *support*, car les autres critères permettant de mesurer la qualité des règles ne sont pas monotones, ce qui a pour conséquence que la valeur du critère pour un ensemble peut être meilleure que pour l'un de ses sous-ensembles, ce qui n'est pas le cas du *support*. Nous avons donc proposé alors une méthode énumérative permettant l'utilisation de plusieurs critères.

Différentes perspectives s'ouvrent sur ce travail. Les premières perspectives que nous pouvons donner concernent les améliorations de nos algorithmes. Nous avons développé durant cette thèse un algorithme dédié au problème de la recherche de règles d'association multi-objectifs. Cet algorithmes peuvent bien sûr être améliorés, notamment au niveau de l'efficacité (le temps de calcul de résolution), mais aussi étendus grâce à l'incorporation d'autres heuristiques ou opérateurs spécifiques aux règles d'association.

Concernant le parallélisme des AGs, les principales perspectives concernent l'étude de différents schémas de parallélisation. De plus, pour le moment chaque île a les mêmes paramètres. Il peut être intéressant de faire des îles différentes, de façon à dédier certaines d'entre elles à l'exploration et d'autres à l'exploitation.

En ce qui concerne l'hybridation avec des méthodes exactes, il est décevant que dans ce contexte (règles d'association multi-objectifs), il ne puisse exister de méthodes exactes plus intéressantes. Cependant il peut être envisageable d'hybrider les méthodes heuristiques avec l'algorithme A-priori, qui dans un sous-espace de recherche pourra trouver toutes les règles vérifiant un seuil sur le *support* et la *confiance*. Ceci pourra, par exemple, permettre de diversifier la recherche. De plus, il serait intéressant de réfléchir à d'autres schémas d'hybridation entre les méthodes. Cependant les perspectives les plus intéressantes concernent le parallélisme, en parallélisant au sein de chaque île les opérateurs de reproduction (opérateurs de croisement et de mutation) et l'opérateur énumératif (méthode exacte) ainsi que l'évaluation (si elle est coûteuse). Il serait aussi intéressant de voir comment l'opérateur exact peut être parallélisé pour le rendre plus rapide et voir comment passer aux Grilles de calcul.

En ce qui concerne les puces à ADN, il est important, pour interpréter les résultats, de prendre en considération les informations disponibles dans d'autres bases de données. Des informations détaillées sont consignées dans des bases publiques telles Gene Ontology Consortium [1] qui vise à annoter les gènes de différents organismes modèles par des termes précisément définies et contrôlées. Le GO consortium fournit une courte description de la fonction moléculaire de chaque protéine, du procédé biologique dans lequel elle est impliquée et sa localisation cellulaire. Il est aussi intéressant de comparer les résultats à ceux issus d'autres chercheurs.

[1] http ://www.geneontology.org

# Bibliographie


[ABB00a] Orly Alter, Patrick O. Brown, and David Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *PNAS*, 97 :10101–10106, 2000.

[ABB00b] M. Ashburner, C.A. Ball, and J. A Blake. Gene ontology : tool for the unification of biology. the gene ontology consortium. *Nat.Genet.*, 25 :25–29, 2000.

[AIP01] F. Angiulli, G. Ianni, and L. Palopoli. On the complexity of inducing categorical and quantitative association rules, 2001.

[AIS93] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

[AK89] E. Aarts and J. Korst. *Simulated annealing and Boltzmann machines : a stochastic approach to combinatorial optimization and neural computing.* John Wiley & Sons, Inc., New York, NY, USA, 1989.

[AK01] J. Azé and Y. Kodratoff. Evaluation de la résistance au bruit de quelques mesures d'extraction de règles d'association. *Extraction des connaissances et apprentissage*, 1(4) :143–154, 2001.

[AKC99] F.Ben Abdelaziz, S. Krichen, and J. Chaouachi. *Meta-heuristics : Advances and trends in local search paradigms for optimization*, chapter A hybrid heuristic for multi-objective knapsack problems, pages 205–212. Kluwer Academic Publishers, 1999.

[ALF99] D.L.A. Araujo, H.S. Lopes, and A.A. Freitas. A Parallel Genetic Algorithm for Rule Discovery in Large Databases. In *Proc. 1999 IEEE Systems, Man and Cybernetics Conf.*, volume III, pages 940–945, Tokyo, Japan, October 1999.

[AS94] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th Intl. Conference on Very Large Databases,Santiago, Chile*, Sept 1994.

[AS96] R. Agrawal and J.C. Shafer. Parallel mining of association rules. *Ieee Trans. On Knowledge And Data Engineering*, 8 :962–969, 1996.

[BA05] C.E. Bichot and J.M. Alliot. Optimisation par colonies de fourmis appliqué au découpage de l'espace aérien européen en zones de qualification. In *RIVF2005*, 2005.

[Bas05] M. Basseur. *Conception d'algorithmes coopératifs pour l'optimisation multi-objectif : Application aux problèmes d'ordonnancement de type Flow-shop.* PhD thesis, Université des Sciences et Techniques de Lille1, Juin 2005.

[BB99] PO. Brown and D. Botstein. Exploring the new world of the genome with dna microarrays. *Nature Genetics Supplement*, 21 :33–37, 1999.

[BHQ01] A. Brazma, P. Hingamp, and J. Quackenbush. Minimum information abouta microarray experiment (miame)-toward standards for microarray data. *Nature Genetics*, 29 :365–371, 2001.

[BHS97] B. Bullnheimer, R.F. Hartl, and C. Strauss. Applying the ant system to the vehicle routing problem. In *Proceedings of the second Metaheuristics International Conference (MIC'97)*, Sophia-Antipolis, France, 1997.

[BLDT04] M. Basseur, J. Lemesre, C. Dhaenens, and E.G. Talbi. Cooperation between branch and bound and evolutionary approaches to solve biobjective flow shop. pages 72–86, Brazil, may 2004.

[Bli67] C.I. Bliss. *Statistics in biology*, volume 1. New York : McGraw-Hill, 1967.

[BM94] J. Brans and B. Mareschal. The promethee-gaia decision support system for multicriteria. *Investigation Operativa*, 4 :102–117, 1994.

[BMS97] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets : generalizing association rules to correlations. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pages 265–276, May 1997.

[BMUT97] S. Brin, R. Motwani, J. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proceedings ACM SIGMOD, USA*, pages 255–264, 1997.

[BST02] M. Basseur, F. Seynhaeve, and E-G. Talbi. Design of Multi-objective Evolutionary Algorithms : Application to the Flow-shop Scheduling Problem. In *Congress on Evolutionary Computation CEC'02*, pages 1151–1156, Honolulu, Hawaii, USA, May 2002.

[BW97] P.J. Bentley and J.P. Wakefield. *Soft Computing in Engineering Design and Manufacturing*, chapter Finding acceptable Pareto-optimal solutions using multiobjective genetic algorithms, pages 231–240. Springer Verlag, London, June 1997.

[BWW99] MA. Behr, MA. Wilson, and WP. Gill WP. Comparative genomics of bcg vaccines by whole-genome dna microarray. *Science*, 1520-3 :284, 1999.

[Cah05] S. Cahon. *ParadisEO : Une plate-forme pour la conception et le déploiement de métaheuristiques parallèles hybrides sur clusters et grilles.* PhD thesis, Université des Sciences et Techniques de Lille1, 2005.

[CB91] P. Clark and R. Boswell. Rule induction with CN2 : Some recent improvements. In *Proc. Fifth European Working Session on Learning*, pages 151–163, Berlin, 1991. Springer.

[CCW98] JR. Cho, MJ. Campbell, and EA. Winzeler. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol Cell*, 2 :65–73, 1998.

[CDG+97] J. Chattratichat, J. Darlington, M. Ghanem, Y. Guo, H. Huning, M. Kohlerand J. Sutiwaraphun, HW To, and D. Yang. Large scale data mining : Challenges and responses. In *Proceedings of the 3th International Conference on Knowledge Discovery and Data Mining*, pages 143–146, August 1997.

[CLSL00] P. Collet, E. Lutton, M. Schoenauer, and J. Louchet. Take it easea. In *PPSN*, pages 891–901, 2000.

[CMM90] R.L. Carraway, T.L. Morin, and H. Moskowitz. Generalized dynamic programming for multicriteria optimization. *European Journal of Operational Research*, 44 :95–104, 1990.

[CNFF96] D.W. Cheung, V.T. Ng, A.W. Fu, and Y.J. Fu. A fast distributed algorithm for mining association rules. In *PDIS : International Conference on Parallel and Distributed Information Systems*. IEEE Computer Society Technical Committee on Data Engineering, and ACM SIGMOD, 1996.

[Coe98a] C.A.C. Coello. An updated survey of ga-based multiobjective optimization techniques. Technical Report RD-98-08, Laboratorio Nacional de Informática Avanzada (LANIA), Xalapa, Veracruz, México, December 1998.

[Coe98b] C.A.C. Coello. Using the min-max method to solve multiobjective optimization problems with genetic algorithms. In *IBERAMIA'98*, LNCS. Springer-Verlag, 1998.

[Col88] N.E. Collins. Simulated annealing - an annotated bibliography. *Am. J. Math. Manage. Sci.*, 8(3-4) :209–307, 1988.

[CS02] Y. Collette and P. Siarry. *Optimisation multiobjectif*. Eyrolles, 2002.

[CVH02] O. Cordon, I. F. Viana, and F. Herrera. Analysis of the best-worst ant system and its variants on the qap. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Proceedings of the Third International Workshop on Ant Algorithms (ANTS'2002)*, volume 2463 of *Lecture Notes in Computer Science*, pages 228–234, Brussels, Belgium, September 12-14 2002. Springer Verlag.

[DAPM00] Kalyanmoy Deb, Samir Agrawal, Amrit Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization : NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[DDG04] G. Di Caro, F. Ducatelle, and L.M. Gambardella. AntHocNet : an ant-based hybrid routing algorithm for mobile adhoc networks. In X. Yao, E. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guervós, J.A. Bullinaria, J. Rowe, P. Tino, A. Kabán, and H.P. Schwefel, editors, *Proceedings of the 8th International Conference on Parallel ProblemSolving from Nature (PPSN VIII)*, volume 3242 of *Lecture Notes in Computer Science*, pages 461–470, Birmingham, UK, September 18-22 2004. Springer-Verlag.

[DEC02] Société DECISIA. *SPAD 5.5, Système Pour l'Analyse des Données*. Pantin France, 2002.

[DFS02] S. Dudoit, J. Fridlyand, and T.P. Speed. Comparaison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Stastical Association*, 97 :77–87, 2002.

[DG89] J.L Deneubourg and S. Goss. Collective patterns and decision-making. *Ethology and Evolution*, pages 295–311, 1989.

[DG01a] K. Deb and T. Goel. Controlled elitist eon-dominated sorting genetic algorithms for better convergence. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 67–81. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[DG01b] Kalyanmoy Deb and Tushar Goel. Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 67–81. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[DJL95] G. Dahl, K. Jornsten, and A. Lokketangen. A tabu search approach to the channel minimization problem. In G. Liu, K-H. Phua, J. Ma, J. Xu, F. Gu, and C. He, editors, *Optimization - Techniques and Applications, ICOTA'95*, volume 1, pages 369–377, Chengdu, China, 1995. World Scientific.

[DK01] K. Deb and D. Kalyanmoy. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[DMC91] M. Dorigo, V. Maniezzo, and A. Colorni. Positive feedback as a search strategy. Technical report, Technical Report 91016, Dipartimento di Elettronica e Informatica, Politecnico di Milano, italie, 1991.

[DMC96] M. Dorigo, V. Maniezzo, and A. Colorni. The Ant System : Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B : Cybernetics*, 26(1) :29–41, 1996.

[Dor92] M. Dorigo. *Learning and Natural Algorithms (in Italian)*. PhD thesis, DEI, Politecnico di Milano,Italy, 1992.

[DPV83] J.L. Deneubourg, J.M. Pasteels, and J.C. Verhaeghe. Probabilistic behaviour in ants : a strategy of errors? *Journal of Theoretical Biology*, 105 :259–271, 1983.

[DVI97] JL. DeRisi and PO. Brown VR. Iyer. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 680-6 :278, 1997.

[Edg81] F. Y. Edgeworth. *Mathematical physics*. P. Keagan, London, England, 1881.

[EDL02] R. Edgar, M. Domrache, and A. E. Lash. Gene expression omnibus : Ncbi gene expression and hybridization array data repository. *Nucleic Acids Research*, 30 :207–210, 2002.

[EG00] M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum*, 22 :425–460, 2000.

[ESBB98] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. In *Proceedings of National Academy of Sciences*, volume 95(25), pages 14863–8, 1998.

[FCD+00] T.S. Furey, N. Christianini, N. Dud'y, D.W. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16, no.10 :906Ű914, 2000.

[FF93] Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization : Formulation, Discussion and Generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

[FF95a] Carlos M. Fonseca and Peter J. Fleming. An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3(1) :1–16, Spring 1995.

[FF95b] C.M. Fonseca and P.J. Fleming. Multiobjective genetic algorithms made easy : Selection, sharing and mating restrictions. In *IEEE Int. Conf. on Genetic Algorithms in Engineering Systems : Innovations and Applications*, pages 45–52, Sheffield, UK, 1995.

[FF95c] C.M. Fonseca and P.J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1) :1–16, 1995.

[Fon95] C.M. Fonseca. *Multiobjective genetic algorithms with applications to control engineering problems.* PhD thesis, University of Sheffield, 1995.

[Fou85] M. P. Fourman. Compaction of symbolic layout using genetic algorithm s. In *Proceedings of the first international conference on genetic algorithms (ICGA)*, pages 141–153, 1985.

[FPSS96] U.M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge discovery and data mining : Towards a unifying framework. In *Knowledge Discovery and Data Mining*, pages 82–88, 1996.

[FR94] T.A. Féo and M.G.C. Resende. Greedy randomized adaptative search procedures. *Journal of Global Optimization*, 6 :109–133, 1994.

[Fre99] A. Freitas. On rule interestingness measures. *Knowledge-Based Systems journal*, 1999.

[Gar99] G. Gardarin. *Internet/intranet et bases de données.* Eyrolles Informatique, 1999.

[GJ79] M.R. Garey and D.S. Johnson. Computer & intractability : A guide to the theory of np-completeness. *CA : W. H. Freeman*, Nov. 1979.

[GKL95] F. Glover, J.P. Kelly, and M. Laguna. Genetic algorithms and tabu search : hybrids for optimization. *Comput. Oper. Res.*, 22(1) :111–134, 1995.

[GL93] F. Glover and M. Laguna. Tabu search. In C. Reeves, editor, *Modern Heuristic Techniques for Combinatorial Problems*, Oxford, England, 1993. Blackwell Scientific Publishing.

[GL98] F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[GLCM94] X. Gandibleux, G. Libert, E. Cartignies, and P. Millot. SMART : étude de la faisabilité d'un solveur de problèmes de mobilisation de réserve tertiaire. *Revue des systèmes de Décision*, 3(1) :45–67, 1994.

[Glo77] F. Glover. Heuristics for integer programming using surrogate constraints. *Decision Sciences*, 8(1) :156 – 166, 1977.

[Glo89a] F. Glover. Tabu search : part i. *ORSA J. on Computing*, 1(3) :190–206, 1989.

[Glo89b] F. Glover. Tabu search : Part ii. *ORSA J. on Computing*, 2(1) :4–32, 1989.

[Glo98] F. Glover. A template for scatter search and path relinking. In *AE '97 : Selected Papers from the Third European Conference on Artificial Evolution*, pages 3–54, London, UK, 1998. Springer-Verlag.

[GMF96] X. Gandibleux, N. Mezdaoui, and A. Freville. A tabu search procedure to solve multi-objective combinatorial optimization problems. In R. Caballero, F. Ruiz, and R. Steuer, editors, *Second Int. Conf. on Multi-Objective Programming and Goal Programming MOPGP'96*, pages 291–300, Torremolinos, Spain, May 1996. Springer-Verlag.

[Gol89] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[GR87a] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Second Int. Conf. on Genetic Algorithms ICGA'2*, pages 41–49, NJ, 1987. Lawrence Erlbaum.

[GR87b] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 41–49, Mahwah, NJ, USA, 1987. Lawrence Erlbaum Associates, Inc.

[GS88] R. M. Goodman and P. Smyth. Information-theoretic rule induction. In *Proceedings of the 1988 European Conference on Artificial Intelligence*, London, 1988. Pitman.

[GS91] R. M. Goodman and P. Smyth. Rule induction using information theory. In *Knowledge Discovery in Databases*, pages 159–176. PAAAI/MIT Press, 1991.

[Gue73] W.C. Guenther. *Concepts of Statistical Inference.* McGraw- Hill, 2nd ed., New York, 1973.

[GWBV02] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3) :389–422, 2002.

[HBC+96] JG. Hacia, LC. Brody, MS. Chee, SP. Fodor, and FS. Collins. Detection of heterozygous muta-tions in brca1 using high density oligonu-cleotide arrays and two-colour fluorescenceanalysis. *Nat.Genet.*, 14 :441–7, 1996.

[HH99] R. Hilderman and H. Hamilton. Knowledge discovery and interestingness measures : A survey, technical report cs 99-04. Technical report, Department of Computer Science,University of Regina, October 1999.

[HH01] R. Hilderman and H. Hamilton. Evaluation of interestingness measures for ranking discovered knowledge. *Lecture Notes in Computer Science*, 2035 :247–259, 2001.

[HJRF94] A. Hertz, B. Jaumard, C.C. Ribeiro, and W.P. Formosinho Filho. A multi-criteria tabu search approach to cell formation problems in group technology with multiple objectives. *RAIRO Recherche Opérationnelle / Operations Research*, 28(3) :303–328, 1994.

[HM79] C.L. Hwang and A.S.M. Masud. Multiple objective decision making - methods and applications. In *Lectures Notes in Economics and Mathematical Systems*, volume 164. Springer-Verlag, Berlin, 1979.

[HMM+00] N.S. Holter, M. Mitra, A. Maritan, M. Cieplak, J.R. Banavar, and N.V. Fedoroff. Fundamental patterns underlying gene expression profiles : Simplicity from complexity. *PNAS*, 97 :8409 – 8414, 2000.

[Hol75a] J. Holland. *Adaptation in natural and artificial systems.* The University of Michigan Press, Ann Arbor, MI, USA, 1975.

[Hol75b] J.H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.

[HPY00] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In Weidong Chen, Jeffrey Naughton, and Philip A. Bernstein, editors, *2000 ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12. ACM Press, 05 2000.

[HS95] M. Houtsma and A. Swami. Set-oriented mining for association rules in relational databases. In P. S. Yu and A. L. P. Chen, editors, *Proceedings of the 11th International Conference on Data Engineering*, pages 25–34, Los Alamitos, CA, USA, mar 1995. IEEE Computer Society Press.

[HTK97] H. Han, C.W. Tseng, and P. Keleher. Reducing synchronization overhead for compiler-parallelized codes on software DSMs. In *Languages and Compilers for Parallel Computing*, 1997.

[HV02] M. Halkidi and M. Vazirgiannis. An introduction to quality assessment in data mining. In *PKDD 2002 Conference*, Finland, 2002.

[HWC00] T.-P. Hong, H.-S. Wang, and W.-C. Chen. Simultaneous applying multiple mutation operators in genetic algorithm. *Journal of Heuristics*, 6(4) :439–455, September 2000.

[HWOS97] D. Halhal, G.A. Walters, D. Ouazar, and D.A. Savic. Water network rehabilitation with a structured messy genetic algorithm. *Journal of Water Resources Planning and Management*, 123(3) :137–146, 1997.

[ID90] A. Inselberg and B. Dimsdale. Parallel coordinates : a tool for visualizing multi-dimensional geometry. In *VIS '90 : Proceedings of the 1st conference on Visualization '90*, pages 361–378, Los Alamitos, CA, USA, 1990. IEEE Computer Society Press.

[IER99] VR. Iyer, MB. Eisen, and DT. Ross. The transcriptional program in the response of human fibroblasts to serum. *Science*, 83-7 :283, 1999.

[IIiT+03] Kazuho Ikeo, Jun Ishi-i, Takurou Tamura, Takashi Gojobori, and Yoshio Taten. Cibex : Center for information biology gene expression database. *C. R. Biologies*, 326 :1079–1082, 2003.

[IMT95] Hisao Ishibuchi, Tadahiko Murata, and I. B. Tursken. Selecting Linguistic Classification Rules by Two-Objective Genetic Algorithms. In *Proceedings of the 1995 IEEE International Conference on Systems, Man and Cybernetics*, pages 1410–1415, Vancouver, Canada, October 1995. IEEE.

[Inc03] SPSS Inc. *SPSS base 10.0 user's guide.* SARL. France, 2003.

[JKDT05] L. Jourdan, M. Khabzaoui, C. Dhaenens, and E-G. Talbi. *A hybrid metaheuristic for knowledge discovery in microarray experiments*, chapter 28, pages 489–506. In Handbook of Bioinspired Algorithms and Applications, CRC Press, USA, October 2005.

[Jon75] K.A. De Jong. *An analysis of the behaviour of a class of genetic adaptive systems.* PhD thesis, University of Michigan, 1975.

[KAJ94] C. Koulamas, S.R. Antony, and R. Jaen. A survey of simulated annealing applications to operations research problems. *Omega International Journal of Management Science*, 22 :41–56, 1994.

[KC02] J.D. Knowles and D.W. Corne. On metrics for comparing non-dominated sets. In IEEE Service Center, editor, *Congress on Evolutionary Computation (CEC'2002)*, volume 1, pages 711–716, Piscataway, New Jersey, May 2002.

[KDNT03] M. Khabzaoui, C. Dhaenens, A. N'Guessan, and E-G. Talbi. Etude exploratoire des critères de qualité des règles d'association en datamining. In *Journées Françaises de Statistique*, pages 583–587, 2003.

[KDT03a] M. Khabzaoui, C. Dhaenens, and E-G. Talbi. Approche évolutionnaire multicritère pout les règles d'association en génomique. In *Cinquième Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la décision ROADEF'2003*, pages 173–174, Avignon, France, 2003.

[KDT03b] M. Khabzaoui, C. Dhaenens, and E-G. Talbi. A multicriteria genetic algorithm for dna microarray data. In *Workshop on Real-life applications of Metaheuristics*, Antwerp, Belgique, December 2003.

[KDT04a] M. Khabzaoui, C. Dhaenens, and E-G Talbi. Association rules discovery for dna microarray data. In *SIAM Bioinformatics Workshop, in conjunction with fourth SIAM International Conference on DataMining*, pages 63–71, Orlando, USA, 24 April 2004.

[KDT04b] M. Khabzaoui, C. Dhaenens, and E-G Talbi. A multicriteria genetic algorithm to analyze dna microarray data. In *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*, pages 1874–1881, Portland, Oregon, 20-23 June 2004. IEEE Press.

[KDT05a] M. Khabzaoui, C. Dhaenens, and E-G. Talbi. Métaheuristique parallèle pour les règles d'association multicritères. In *Sixième congrès de la Société Française de Recherche Opérationnelle et Aide à la décision (ROADEF'2005)*, pages 233–234, Tours, France, February 2005.

[KDT05b] M. Khabzaoui, C. Dhaenens, and E-G Talbi. Parallel genetic algorithms for multi-objective rule mining. In *Proceedings of the sixteenth Metaheuristic International Conference (MIC 2005)*, pages 571–576, Vienna, Autria, 22-26 August 2005.

[KDT06a] M. Khabzaoui, C. Dhaenens, and E-G. Talbi. Combining evolutionary algorithms and exact approaches for multi-objective knowledge discovery. *RAIRO Operations Research, Special Issue on Cooperative methods for Multiobjective Optimization*, accepted, 2006.

[KDT06b] M. Khabzaoui, C. Dhaenens, and E-G. Talbi. *A Cooperative Genetic Algorithm for Knowledge Discovery in MicroArray Experiments*, chapter 13, pages 305–326. 0-471-71848-3. In Parallel Computing for Bioinformatics and Computational Biology, April 2006.

[KGV83] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 220, 4598 :671–680, 1983.

[kJ95] G. kontoravdis and J.F.Bard. Improved heuristics for the vehicle routing problem with time windows. *ORSA Journal on Computing*, 7 :10–23, 1995.

[KJK+01] S.K. Kim, J.Lund, M. Kiraly, K. Duke, M. Jiang, J.M. Stuart, A. Eizinger, B.N. Wylie, and G.S. Davidson. A gene expression map for caenorhabditis elegans. *Science*, 293 :2087 – 2092, 2001.

[KMRS01] M. Keijzer, J.J. Merelo, G. Romero, and M. Schoenauer. Evolving objects : A general purpose evolutionary computation library. In *Artificial Evolution*, pages 231–244, 2001.

[Kon94] K. Konolige. Easy to be hard : Difficult problems for greedy algorithms. *Proc.of 4th Intl. Conf. on Principles of Knowledge Representation and Reasoning*, pages 374–378, 1994.

[Kur91] F. Kursawe. A variant of evolution strategies for vector optimization. In H.-P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature*, pages 193–197. Springer, 1991.

[LBF98] X. Liu, D. Begg, and R.J. Fishwick. Genetic approach to optimal topology/-controller design of adaptive structures. *International Journal for Numerical Methods in Engineering*, 41 :815–830, 1998.

[LDB+] D.J. Lockhart, H. Dong, M.C. Byrne, M.T. Follettie, M.V. Gallo, M.S. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Horton, and E.L. Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 14 :1675–1680.

[LDT05] J. Lemesre, C. Dhaenens, and E.-G. Talbi. An exact parallel method for a bi-objective permutation flowshop problem. *European Journal of Operational Research*, 2005. to appear.

[LE97] M. A. Lee and H. Esbensen. Fuzzy/Multiobjective Genetic Systems for Intelligent Systems Design Tools and Components. In Witold Pedrycz, editor, *Fuzzy Evolutionary Computation*, pages 57–80. Kluwer Academic Publishers, Boston, Massachusetts, 1997.

[LER81] I.C. LERMAN. *Classification et analyse ordinale des données.* Dunod, 1981.

[LFE94] M. Laguna, T.A. Feo, and H.C. Elrod. A greedy randomized adaptive search procedure for the two-partition problem. *Operations Research*, 42 :677–687, 1994.

[LFZ99] N. Lavrač, P. Flach, and B. Zupan. Rule evaluation measures : A unifying view. In S. Džeroski and P. Flach, editors, *ILP99*, volume 1634 of *LNAI*, pages 174–185. SV, 1999.

[LMP95] L. Lebart, A. Morineau, and M. Piron. *Statistique Exploratoire Multidimensionnelle.* Dunod, 1995.

[Loe47] J. Loevinger. A systemic approach to the construction and evaluation of tests of ability. In *Psychological monographs*, volume 61(4), 1947.

[LT03] S. Lallich and O. Teytaud. Evaluation et validation de l'intérêt des règles d'association. *Revue des Nouvelles Technologies de l'Information*, (2), 2003.

[MBdSD04] K.K. Mantripragada, P.G. Buckley, T.D. de Stahl, and J.P. Dumanski. Genomic microarrays in the spotlight. *Trends in genetics*, 20(2) :87–94, 2004.

[MDB98] MJ. Marton, JL. DeRisi, and HA. Bennett. Drug target validation and identification of secondary drug target effects using dna microarrays. *Nat Genet*, 1293-301 :4, 1998.

[MM96] L. Mandow and E. Millan. Goal programming and heuristic search. In R. Caballero, F. Ruiz, and R. Steuer, editors, *Second Int. Conf. on Multi-Objective Programming and Goal Programming MOPGP'96*, pages 48–56, Torremolinos, Spain, May 1996. Springer-Verlag.

[MMZ+04] SA. McCarroll, CT. Murphy, S. Zou, SD. Pletcher, CS. Chin, YN. Jan, C. Kenyon, CI. Bargmann, and H. Li. Comparing genomic expression patterns across species identifies shared transcriptional profile in aging. *Nat.Genet.*, 36 :197–204, 2004.

[MPT00] S. Mardle, S. Pascoes, and M. Tamiz. An investigation of genetic algorithm for the optimization of multi-objective fisheries bioeconomic models. *International Transaction of Operations Research*, 7 :33–49, 2000.

[MTR00] H. Meunier, E. G. Talbi, and P. Reininger. A multiobjective genetic algorithm for radio network optimisation. In *CEC*, volume 1, pages 317–324, Piscataway, New Jersey, July 2000. IEEE Service Center.

[MTV94] H. Mannila, H. Toivonen, and A.I. Verkamo. Efficient algorithms for discovering association rules. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, *AAAI Workshop on Knowledge Discovery in Databases (KDD-94)*, pages 181–192, Seattle, Washington, jul 1994. AAAI Press.

[NHBM98] D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz. UCI repository of machine learning databases, 1998.

[NPL99] R. Taouil N. Pasquier, Y. Bastide and L. Lakhal. Efficient mining of association rules using closed itemset lattices. *Information Systems*, 24(1) :25–46, 1999.

[Par96] V. Pareto. *Cours d'Economie Politique.* Rouge, Lausanne, Switzerland, 1896.

[PCY95] J.S. Park, M-S. Chen, and P.S. Yu. An effective hash based algorithm for mining association rules. In Michael J. Carey and Donovan A. Schneider, editors, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 175–186, San Jose, California, 22–25 1995.

[Pea88] J. Pearl. Probabilistic reasoning in intelligent systems. *Morgan Kaufmann*, 1988.

[PL03] P. Picouet et B. Vaillant P. Lenca, P. Meyer. Aide multicritère à la décision pour évaluer les indices de qualité des connaissances. In RSTI-RIA, editor, *Extraction et Gestion des Connaissances*, volume 17/2003, pages 271–282. 2003.

[PLF02] R.S. Parpinelli, H.S. Lopes, and A.A. Freitas. *Data Mining : a Heuristic Approach*, chapter An Ant Colony Algorithm for Classification Rule Discovery, pages 191–208. London : Idea Group Publishing, 2002.

[PM98a] Geoffrey T. Parks and I. Miller. Selective Breeding in a Multiobjective Genetic Algorithm. In A. E. Eiben, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving From Nature — PPSN V*, pages 250–259, Amsterdam, Holland, 1998. Springer-Verlag.

[PM98b] G.T. Parks and I. Miller. Selective breeding in a multiobjective genetic algorithm. In *Parallel Problem Solving from Nature PPSN'5*, pages 250–259, Amsterdam, September 1998. Springer-Verkag.

[PR02] L.S. Pitsoulis and M.G.C. Resende. Greedy randomized adaptive search procedures. In P.M. Pardalos and M.G.C. Resende, editors, *Handbook of Applied Optimization*, pages 178–183. Oxford University Press, 2002.

[PS82] C.H. Papadimitriou and K. Steiglitz. *Combinatorial optimization : algorithms and complexity.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.

[PS91] G. Piatetsky-Shapiro. Discovery, analysis and presentation of strong rules. In *Knowledge Discovery in Databases, AAAI/MIT Press*, pages 229–248, 1991.

[PSS+04] H. Parkinson, U. Sarkans, M. Shojatalab, N. Abeygunawardena, S. Contrino, R. Coulson, A. Farne, G. Garcia Lara, E. Holloway, M. Kapushesky, P. Lilja, G. Mukherjee, A. Oezcimen, T. Rayner, P. Rocca-Serra, A. Sharma, S. Sansone, and A. Brazma. Arrayexpress-a public repository for microarray gene expression data at the ebi. *Nucleic Acids Research*, 33 :D553–D555, 2004.

[PWCG01] P. Pavlidis, J. Weston, J. Cai, and W.N. Grundy. Gene functional classification from heterogenous data. In *Proceedings of the Fifth Annual International Conference on Computational Biology (RECOMB)*, pages 249–255, 2001.

[RB98] J. Roberto and Jr. Bayardo. Efficiently mining long patterns from databases. In Laura M. Haas and Ashutosh Tiwary, editors, *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 85–93. ACM Press, 1998.

[RB04] S. Rojanasoonthon and J.F. Bard. A grasp for parallel machine scheduling with time windows. *INFORMS Journal on Computing*, 2004. To appear.

[RBR03] S. Rojanasoonthon, J.F. Bard, and S.D. Reddy. Algorithms for parallel machine scheduling : A case study of the tracking and data relay satellite system. *Journal of the Operational Research Society*, 54 :806–821, 2003.

[RER94] B.J. Ritzel, J.W. Eheart, and S. Ranjithan. Using genetic algorithms to solve a multiple objective groundwater pollution problem. *Water Resources Research*, 30(5) :1589–1603, May 1994.

[Roy85] B. Roy. *Méthodologie multicritère d'aide à la décision.* Economica, Paris, 1985.

[Sch85] J.D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In J.J. Grefenstette, editor, *ICGA Int. Conf. on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum, 1985.

[Sch95] J.R. Schott. *Fault tolerant design using single and multicriteria genetic algorithm optimization.* PhD thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technologie, Cambridge, Massachusetts, USA, 1995.

[SD95a] N. Srinivas and K. Deb. Multiobjective optimisation using non-dominated sorting in genetic algorithms. *Evolutionary Computation*, 2(8) :221–248, 1995.

[SD95b] N. Srinivas and Kalyanmoy Deb. Comparative study of vector evaluated GA and NSGA applied to multiobjective optimization. In P. K. Roy and S. D. Mehta, editors, *Proceedings of the Symposium on Genetic Algorithms*, pages 83–90, 1995.

[SDP01] V. Shi, S. Duanmu, and W. Perrizo. Principles for measuring association rules. Technical report, Computer Science Department, North Dakota State University, 2001.

[SK96] T. Shintani and M. Kitsuregawa. Hash based parallel algorithms for mining association rules. In *PDIS : International Conference on Parallel and Distributed Information Systems.* IEEE Computer Society Technical Committee on Data Engineering, and ACM SIGMOD, 1996.

[SK99] S. Sayin and S. Karabati. A bicriteria approach to the two-machine flow shop scheduling problem. *European Journal of Operational Research*, 113 :435–449, 1999.

[Smi83] S. Smith. Flexible learning of problem solving heuristics through adaptive search. In *Proceedings 8th International Joint Conference on Artificial Intelligence*, August 1983.

[SON95] A. Savasere, E. Omiecinski, and S.B. Navathe. An efficient algorithm for mining association rules in large databases. In *VLDB '95 : Proceedings of the 21th International Conference on Very Large Data Bases*, pages 432–444, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[Spi72] C.C. Spicer. *Algorithm AS 52 : Calculation of power sums of deviations about the mean, Applied Statistics,21.* 1972.

[SRD88] T. Sen, M.E. Raiszadeh, and P. Dileepan. A branch and bound approach to the bicriterion scheduling problem involving total flowtime and range of lateness. *Management Science*, 34(2) :254–260, 1988.

[SS88] M. Sebag and M. Schoenauer. Generation of rules with certainty and confidence factors from incomplete and incoherent learning bases. In J. Boose,

B. Gaines, and M. Linster, editors, *Proc. of the European Knowledge Acquisition Workshop (EKAW'88)*, pages 28–1 – 28–20. Gesellschaft für Mathematik und Datenverarbeitung mbH, Sankt Augustin, Germany, 1988.

[SSDB95] Mark Schena, Dari Shalon, Ronald W. Davis, and Patrick O. Brown. Quantitative monitoring of gene expression patterns with a complementary dna microarray. *Science*, 270 :467–470, 1995.

[SSKK03] Joshua M. Stuart, Eran Segal, Daphne Koller, and Stuart K. Kim. A gene co-expression network for global discovery of conserved genetic modules. *Science*, 302 :249–255, 2003.

[STVC+02] L.H. Saal, C. Troein, J. Vallon-Christersson, S. Gruvberger, A. Borg, and C. Peterson. Bioarray software environment (base) : a platform for comprehensive management and analysis of microarray data. *Genome Biology*, 3 :software003, 2002.

[SW91] B.S. Stewart and C.C. White. Multiobjective a*. *Journal of the ACM*, 38(4) :775–814, 1991.

[Tal00] E-G. Talbi. Metaheuristics for multiobjective combinatorial optimization : state of the art. Technical report, LIFL, University of Lille, France, 2000.

[Tal02] El-Ghazali Talbi. A taxonomy of hybrid metaheuristics. *J. Heuristics*, 8(5) :541–564, 2002.

[TKS02] P-N. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *Proceedings of the Eight ACM SIGKDD conference, Edmonton, Canada*, 2002.

[TMA+95] H. Tamaki, M. Mori, M. Araki, Y. Mishima, and H. Ogai. Multi-Criteria Optimization by Genetic Algorithms : A Case of Scheduling in Hot Rolling Process. In *Proceedings of the 3rd Conference of the Association of Asian-Pacific Operational Research Societies within IFORS (APORS'94)*, pages 374–381. World Scientific, 1995.

[Toi96] H. Toivonen. Sampling large databases for association rules. In T. M. Vijayaraman, Alejandro P. Buchmann, C. Mohan, and Nandlal L. Sarda, editors, *In Proc. 1996 Int. Conf. Very Large Data Bases*, pages 134–145. Morgan Kaufman, 09 1996.

[Ulu93] E.L. Ulungu. *Optimisation combinatoire multicritère : détermination de l'ensemble des solutions efficaces et méthodes intéractives.* PhD thesis, Université de Mons-Hainaut, 1993.

[UT94] E.L. Ulungu and J. Teghem. Multi-objective combinatorial optimization : a survey. *Journal of Multi-Criteria Decision Analysis*, 3 :83–104, 1994.

[UT95] E.L. Ulungu and J. Teghem. The two phase method : An efficient procedure to solve bi-objective combinatorial optimization problems. In *Foundations of Computing and Decision Sciences*, volume 20, pages 149–165. 1995.

[UTFT98] E.L. Ulungu, J. Teghem, P. Fortemps, and D. Tuyttens. MOSA method : A tool for solving multi-objective combinatorial optimization problems. Technical report, Laboratory of Mathematic and Operational Research, Faculté Polytechnique de Mons, 1998.

[VL00] D. A. Van Veldhuizen and G. B. Lamont. On measuring multiobjective evolutionary algorithm performance. In *In 2000 Congress on Evolutionary Computation, Piscataway, New Jersey*, volume 1, pages 204–211, July 2000.

[VSM+97] D.A. Van Veldhuizen, B.S. Sandlin, R.E. Marmelstein, G.B. Lamont, and A.J. Terzuoli. Finding improved wire-antenna geometries with genetic algorithms. In P.K. Chawdhry, R. Roy, and P.K. Pant, editors, *Soft Computing in Engineering Design and Manufacturing*, pages 231–240, London, June 1997. Springer Verlag.

[VTPU98] M. Visée, J. Teghem, M. Pirlot, and E.L. Ulungu. Two-phases method and branch and bound procedures to solve knapsack problem. *Journal of Global Optimization*, 12 :139–155, 1998.

[VZVK95] V.E. Velculescu, L. Zhang, B. Vogelstein, and K.W. Kinzler. Serial analysis of gene expression. *Science*, 270 :484–487, 1995.

[WFS98] Wang, JB. Fan, and CJ. Siao. Large-scale identification, mapping, and genoty-ping of single-nucleotide polymorphisms inthe human genome. *Science*, 280 :1077–82, 1998.

[Whi82] D.J. White. The set of efficient solutions for multiple-objectives shortest path problems. *Computers and Operations Research*, 9 :101–107, 1982.

[WLK92] P.B. Wienke, C. Lucasius, and G. Kateman. Multicriteria target optimization of analytical procedures using a genetic algorithm. *Analytical Chimica Acta*, 265(2) :211–225, 1992.

[WWT99] P.C. Wong, P. Whitney, and J. Thomas. Visualizing association rules for text mining. In *INFOVIS*, pages 120–123, 1999.

[YDL+02] Y. Yang, S. Dudoit, D. Lin, V. Peng, J. Ngai, and T. Speed. Normalization for cdna microarray data : A robust composite method adressing single and multiple slide systematic variation. *Nat.Genet.*, 30(4) :e15.1–e15.10, 2002.

[Zak99] M.J. Zaki. Parallel and distributed association mining : A survey. *IEEE Concurrency*, 7(4) :14–25, /1999.

[ZG99] G. Zhou and M. Gen. Genetic algorithm approach on multi-criteria minimum spanning tree problem. *European Journal of Operational Research*, 114 :141–152, 1999.

[Zit99] E. Zitzler. Evolutionary alg orithms for multiobjective optimization : Methods and applications. Master's thesis, Swiss federal Institute of technology (ETH), Zurich, Switzerland, November 1999.

[ZOPL96] M.J. Zaki, M. Ogihara, S. Parthasarathy, and W. Li. Parallel data mining for association rules on shared-memory multiprocessors. Technical Report TR618, 1996.

[ZPOL97] M.J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. Technical Report TR651, 1997.

[ZT98] Eckart Zitzler and Lothar Thiele. An Evolutionary Algorithm for Multiobjective Optimization : The Strength Pareto Approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, May 1998.

[ZT99] Eckart Zitzler and Lothar Thiele. Multiobjective Evolutionary Algorithms : A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4) :257–271, November 1999.