UNIVERSITÉ FRANÇOIS RABELAIS
TOURS

École Doctorale : Santé, Sciences
et Technologies

Année Universitaire : 2006-2007

THÈSE POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ DE TOURS

Discipline : Informatique

présentée et soutenue publiquement

par :

Alain DELAPLACE

le 12 décembre 2007

Approche évolutionnaire de l'apprentissage de structure pour les réseaux bayésiens

---

Directeur de thèse : Professeur Hubert Cardot

Co-encadrement : Thierry Brouard

JURY :

Thierry BROUARD	*Examinateur*	Maître de Conférences à l'Université de Tours
Hubert CARDOT	*Directeur de thèse*	Professeur des universités à l'Université de Tours
Philippe LERAY	*Rapporteur*	Professeur des universités à l'Université de Nantes
Jaime LOPEZ KRAHE	*Examinateur*	Professeur des universités à l'Université Paris 8
Michèle SEBAG	*Rapporteur*	Directeur de Recherches à l'Université Paris-Sud

UNIVERSITÉ FRANÇOIS RABELAIS
TOURS

École Doctorale : Santé, Sciences
et Technologies

Année Universitaire : 2006-2007

THÈSE POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ DE TOURS

Discipline : Informatique

présentée et soutenue publiquement

par :

Alain DELAPLACE

le 12 décembre 2007

Approche évolutionnaire de l'apprentissage de structure pour les réseaux bayésiens

---

Directeur de thèse : Professeur Hubert Cardot

Co-encadrement : Thierry Brouard

JURY :

Thierry BROUARD	*Examinateur*	Maître de Conférences à l'Université de Tours
Hubert CARDOT	*Directeur de thèse*	Professeur des universités à l'Université de Tours
Philippe LERAY	*Rapporteur*	Professeur des universités à l'Université de Nantes
Jaime LOPEZ KRAHE	*Examinateur*	Professeur des universités à l'Université Paris 8
Michèle SEBAG	*Rapporteur*	Directeur de Recherches à l'Université Paris-Sud

"There's no such thing as a free lunch."
"Specialization is for insects."
Robert A. Heinlein

Remerciements

Je tiens à remercier en premier lieu, pour la place qu'ils ont eu dans ces années de travail, Hubert Cardot et Thierry Brouard. Je les remercie de la liberté de choix et de recherche qu'ils m'ont laissé tout en sachant me guider une fois mes choix décidés. Je tiens aussi par ailleurs à m'excuser auprès d'eux pour les relectures, nombreuses, et certainement pas toujours agréables étant donné mon amour des grandes phrases.

Je tiens aussi remercier mes rapporteurs. Philippe Leray, pour avoir su, que ce soit directement ou indirectement, me conseiller et m'avoir fait découvrir le domaine des réseaux bayésiens. Je ne pense pas être le seul dans ce cas étant donné la popularité grandissante des réseaux bayésiens dans nos provinces. Merci, de même, à Michèle Sebag pour ses remarques concernant les différents points de mon travail autour des méthodes évolutionnaires et avoir su partager son expérience. Le travail final en a été grandement amélioré.

Mes parents, bien sûr. C'est évident, bien sûr, mais c'est à vous que je dois d'en être là. Entre les encouragements, le toit et les repas chauds ainsi que les coups de pieds souvent mérités, vous avez été là du début à la fin et il faudrait être le dernier des ingrats pour ne pas en rendre compte. Du début à la fin, à chaque galère, chaque moment difficile, vous étiez là. De là à penser que vous me portez la poisse... Ma famille, en général, et ma tante Nadia pour m'avoir toujours laissé une gamelle à chauffer à deux heures du matin, quand les barres du distributeur ne suffisaient plus.

Maintenant, la partie désopilante. La section des remerciements aura constitué pour moi une des parties les plus problématiques à rédiger. Qui doit y figurer ? Dans quel ordre et surtout, aurais-je oublié quelqu'un ? Dois-je décevoir mon public qui s'attend à une avalanche de gags et de calembours en une fusion miraculeuse d'un almanach Vermot et d'une section de remerciements traditionnelle, du genre à tirer des larmes à un parpaing ?

Usuellement constituée d'un défilé de sobriquets ridicules évoquant une vie sociale depuis longtemps moribonde et de références appuyées à d'interminables nuits de travail, illustrant la pathétique mais virile nostalgie digne d'une chambrée militaire des heures fichées en l'air sur des sujets stériles, la page des remerciements est traditionnellement à un manuscrit de thèse ce que les dés en mousse sont à une voiture.

Quand des années censées être les plus intenses d'une existence ne semblent avoir connu comme sommets que de tristes soirées tartiflettes passées devant une empoignade télévisée de sombres néanderthaliens en short avec des gens dont on ignore pour la plupart les prénoms, on se fait rapidement une idée de la qualité de vie allant de pair avec les études longues.

Mais puisqu'il faut en passer par là, allons-y.

À mes amis, Mathieu, Guillaume, Clément, Christophe, Ludo... Merci pour avoir supporté mon sale caractère et mon cynisme meurtrier. Mes dépressions n'auraient pas été les mêmes sans vous. Plus sérieusement, il est du domaine public que j'ai un caractère de cochon mais je sais que les amis, ce sont les gens encore là après que l'on se soit comporté comme un crétin.

Les compagnons de galère du labo : Stéphane, le trio Julien O., Julien M. et Ludo P.(les grands musiciens sont ceux aux carrières les plus courtes, je vous souhaite de rester petits), Sébastien D.(lève le pied et dors un peu, à ce point-là ça relève du masochisme), Rashid (celui-qui-ne dors-jamais, ou rarement), Lamia (pour les nuits de rédaction et pour être toi-même, tout simplement), Geoffrey, les Mathieus, Cédric, David, Arnaud, et les dizaines d'autres doctorants passés ou présents. Merci en particulier à Sébastien Aupetit pour son aide sur le domaine des algorithmes évolutionnaires. Je n'aime pas expédier ainsi de ce que je ne saurais considérer comme une "tâche" mais il me faudrait trop de pages pour vous remercier chacun et chacune à la hauteur du/des service(s) rendu(s). Et Raoni me ferait un procès, en plus.

Les doctorants "extérieurs" (bah !) : Olivier François pour son travail sur la BNT et les réseaux bayésiens mais aussi les conseils et tout le reste, Sabine Barrat pour m'avoir montré que mon travail pouvait effectivement servir à d'autres personnes, Nicolas Marti pour... euh, pour les Martineries, Cheng-Ma pour m'avoir aidé à découvrir une culture qui me fascine toujours, et là aussi beaucoup d'autres.

Les personnes travaillant au laboratoire d'informatique de l'université de Tours, bien sûr. Merci à tous ceux qui auront pris le temps, parfois, de passer mon bureau juste pour me demander comment ça allait. Ce n'est pas grand-chose mais, au final, ça compte. Merci bien sûr aux membres de mon équipe, l'équipe RFAI mais aussi à tous ceux des autres équipes pour m'avoir donné le coup de main quand j'en avait besoin et ce, parfois, avant même que je ne commence ma thèse. Un énorme et franc merci à Jean-Charles Billaut, Ameur Soukhal, Vincent T'kindt, Christophe Lenté, Mohand Slimane et en général à tous ceux qui ont pris le temps. Et je n'oublie pas les IATOS sans lesquels on n'irait pas bien loin. Un merci spécial à Colette, qui s'arrange toujours pour que les trains partent à l'heure et que tous les papiers soient bien signés.

Et merci, bien sûr, à Christian Proust pour avoir permis que tout cela soit possible.

Ah oui, merci aussi à Georges, Brad et Angelina pour cette délicieuse soirée au Georges V.

Et un grand, un immense merci à tous ceux qui n'y ont pas cru (ils se reconnaîtront). Vous suprendre aura été une belle motivation pour aller au bout.

Enfin, je concluerai en précisant que la liste des personnes que je remercie ne saurait être exhaustive ; pirouette élégante qui m'évitera de me morfondre, une fois que ce travail sera imprimé et dûment relié, lorsque je me rendrai compte que j'aurai oublié quelqu'un d'important.

Table des matières

Chapitre 1

Introduction

La recherche s'est depuis longtemps penchée sur la restitution ou du moins la simulation du fonctionnement de l'esprit humain. L'une des tentatives les plus reconnues est celle visant à pouvoir simuler le processus d'apprentissage de l'être humain de manière automatique à travers un système théorique. Ce système devant alors être apte à apprendre par l'expérience et, par voie de conséquence, à s'améliorer dans l'exécution de la tâche qui lui a été confiée. L'objectif que nous venons d'énoncer est le principe général de la discipline que l'on nomme *machine learning*. Sous cette dénomination se retrouve effectivement un ensemble de méthodes et de modèles dont l'objectif est de pouvoir extraire et intégrer une connaissance par la voie de l'apprentissage automatique.

L'étendue du champ d'application des modèles graphiques est aussi vaste que la taxonomie s'y rapportant. Si les modèles dirigés s'apprêtent à l'assistance au diagnostic et à l'inférence, d'autres modèles (semi-dirigés ou non dirigés) ont été développés à mesure du temps et des besoins afin de pouvoir, par exemple, s'appliquer à la segmentation d'images, au filtrage de signal, etc.

L'intérêt d'un modèle et de l'apprentissage en général est d'obtenir un système certes capable de se perfectionner à travers sa propre expérience mais aussi apte à s'adapter à des situations différentes. Pour prendre un exemple simple, l'expérience que l'on peut avoir en travaillant dans l'informatique peut servir, à l'occasion, à dépanner un appareil électronique. Cette adaptation à de nouvelles situations à partir de l'acquis est aussi une des caractéristiques de l'humain. Les objectifs ont évolué. Par-delà la simple émulation de la cognition humaine, les modèles et méthodes du *machine learning* ont aujourd'hui réussi à rassembler en leur sein des systèmes dont les objectifs peuvent être aussi divers que :

- extraire une connaissance trop complexe pour pouvoir être décrite par un expert ;
- aider l'expert en lui apportant une connaissance simplifiée ou non du domaine ;
- inversement, pouvoir intégrer une connaissance experte à un domaine ;
- être capable, par un apprentissage incrémental, de restituer le mécanisme sous-jacent à l'objet modélisé ;
- offrir un formalisme universel afin de faciliter la transmission de la connaissance acquise ;
- ...

Le domaine du *machine learning* a considérablement évolué depuis le milieu du XX^e siècle, aussi bien à travers les modèles, depuis les champs de Markov cachés jusqu'aux machines à

vecteurs de support, qu'à travers ses nombreuses applications allant de l'application industrielle et commerciale jusqu'aux applications militaires, à l'aide à la vie courante sous forme électronique ou encore le diagnostic médical.

Conçus pour pouvoir prendre en charge les problèmes comportant la notion d'incertitude, les réseaux bayésiens apportent à la fois une interface intuitive sous la forme d'un graphe orienté et un ensemble de méthodes permettant d'exploiter au mieux la connaissance extraite qu'ils modélisent. Par conséquents, les réseaux bayésiens se sont peu à peu imposés parmi les différents modèles probabilistes existants. Si les réseaux bayésiens ont été connus principalement grâce aux travaux de Judea Pearl [Pearl, 1988] et Michael Jordan [Jordan, 1998], les premières ébauches de ces modèles remontent au début du XX^e siècle avec les travaux de S. Wright [Wright, 1921].

De toutes les problématiques gravitant autour des réseaux bayésiens, la détermination du modèle même est la plus cruciale et la plus étudiée. Si la détermination complète d'un réseau bayésien – ou de tout modèle en général – par un expert paraît être la solution la plus simple, il en est hélas autrement. D'une part, une telle détermination est coûteuse en temps et en moyens. Il est rare de pouvoir promptement déterminer un modèle fiable d'un domaine constitué de nombreuses variables. D'autre part, le modèle obtenu par apprentissage constitue lui-même, dans certains cas, la solution recherchée. On peut ainsi souhaiter déterminer les interactions entre différents allèles d'un gène et donc, à partir d'une base d'exemples, chercher le modèle reflétant au mieux ces relations. Dans ce cas, la réponse (le modèle) est partie intégrante du problème et un expert ne pourrait répondre au besoin.

Diverses méthodologies ont été développées dans le but de permettre un apprentissage automatique des constituants d'un réseau bayésien : méthodes déterminant des relations probabilistes à partir de tests d'indépendance statistique, méthodes élisant le meilleur modèle à partir d'un ensemble de candidats ou encore recherche de la meilleure classe d'équivalence.

En l'occurrence, un type populaire d'heuristique d'apprentissage pouvant se prêter à un tel exercice est l'ensemble des méthodes dites évolutionnaires et plus particulièrement les algorithmes génétiques. Issus, dans leur forme actuelle, des travaux de J. H. Holland [Holland, 1975] dans les années soixante-dix, les algorithmes génétiques partagent avec les réseaux bayésiens un facteur d'attrait non négligeable en ce que leur fonctionnement est intuitif et aisément assimilable. Inspirés des théories de Darwin et de l'idée de sélection naturelle, leur principe de sélection basé sur la qualité d'un individu en fait un type d'heuristique visant à une performance individuelle tout en étant capables, à la différence d'heuristiques exactes, de faire une synthèse des résultats. Un comportement tout à fait comparable aux objectifs du *machine learning*. Les algorithmes génétiques font de plus, depuis un certain nombre d'années, l'objet de plusieurs études visant à les sortir du carcan formaliste les restreignant jusqu'alors à un simple schéma d'exploration stochastique/exploitation. Ces études mettent surtout en relief l'importance de la représentation des solutions ou la possibilité d'automatiser les paramètres de recherche de l'algorithme.

Nous proposons ici d'étudier le comportement et les performances d'un tel algorithme génétique lors de l'apprentissage de la structure d'un réseau bayésien. Nous mettrons en évidence les qualités et les défauts respectifs des différents outils et méthodes développés et employés. Comme cela est souvent le cas dans la littérature, nous nous sommes fixés pour objectif de parvenir à retrouver une structure connue à partir de bases d'exemples préalablement échan-

tillonnés. Nous avons aussi observé le comportement et les performances de réseaux bayésiens appliqués à la classification dans le cadre spécifique de la reconnaissance de formes.

1.1 Problématique et objectifs de notre travail

À la lecture de la littérature, il s'avère que les travaux effectués sur l'emploi de méthodes évolutionnaires pour l'apprentissage de la structure des réseaux bayésiens se sont, pour la plupart, limités à l'application d'un algorithme génétique sous sa forme canonique sur un espace préalablement restreint ou bien à l'espace des structures à l'aide d'opérateurs eux-mêmes restreints. L'objectif de notre travail est de déterminer, à travers plusieurs approches, si les développements ultérieurs des processus évolutionnaires sont à même d'apporter un réel bénéfice à une telle approche du problème. Une première approche, consistant en une méthode de *niching* séquentiel adaptée, exploite les propriétés de l'espace des graphes représentants des classes d'équivalence des structures [Delaplace et al., 2007a, Delaplace et al., 2007b]. Une évolution de cette méthode, conjuguant l'aspect temporel des méthodes séquentielles à une recherche répartie dans l'espace des solutions, applique le même principe à une population répartie en îlots. Enfin, une autre méthode, améliorant et précisant les premiers principes de mutation dynamique appliqués à notre problème et exposés dans [Delaplace et al., 2006], modélise une distribution de probabilités pour les différentes opérations de mutation applicables aux structures évoluées ; distribution réévaluée en fonction des résultats observés au cours des phases successives de mutation.

1.2 Guide de lecture

Dans un premier temps, à travers l'état de l'art, nous aborderons le thème des réseaux bayésiens. Nous présenterons les caractéristiques de cette modélisation avant de voir quelles sont les principales méthodes d'apprentissage des paramètres avant d'aborder les méthodes existantes d'apprentissage des paramètres d'un réseau bayésien à partir d'une base de cas.

Dans une deuxième partie, nous présenterons les différentes stratégies développées dans le cadre de nos travaux. Nous introduirons une adaptation des techniques de *niching* séquentielles à l'apprentissage de structure ainsi qu'une extension de cette méthode par une distribution des individus dans l'espace. Dans le chapitre suivant, une méthode permettant une adaptation de l'opérateur de mutation en fonction des résultats précédemment obtenus sera présentée. Les expérimentations et résultats obtenus à l'aide de ces méthodes ainsi qu'un comparatif avec les principales méthodes de recherche de structure par évaluation existantes seront présentées dans le chapitre 8. Enfin, nous présenterons une application des réseaux bayésiens à la reconnaissance de formes et plus particulièrement à la segmentation de l'iris sur des photographies d'œil humain.

Nous terminerons ce document par une discussion autour des résultats de nos méthodes en détaillant les conclusions que nous avons pu tirer de nos recherches et expérimentations.

En fin de document, outre la bibliographie regroupant les différentes références, le lecteur pourra trouver une annexe contenant les principaux rappels quant à certaines notions em-

ployées dans nos travaux. Parmi celles-ci se trouvent quelques notions de probabilités, une description de certaines techniques de caractérisation de textures employées dans le chapitre dédié à la classification ainsi que les résultats de tests que nous avons effectués dans le cadre du paramétrage de nos algorithmes.

Notations

Notations générales	
$\mathcal{B}$	Réseau bayésien, $B = (\mathcal{G}, \Theta)$.
X_i	Indifféremment, variable aléatoire ou sommet associé dans un graphe.
r_i	Dimension de la variable X_i.
x_i^k	k^e instanciation de la variable aléatoire X_i, $k \in 1 \dots r_i$.
V_i	Liste des r_i instanciations de X_i dans D.
v_{ik}	k^e élément de V_i.
$\mathcal{U}$	Ensemble de n variables aléatoires $\{X_1, X_2, \dots X_n\}$.
D	Base de cas issus du domaine $\mathcal{U}$, constituée de N cas.
Θ	Ensemble des paramètres d'un réseau bayésien $\mathcal{B}$.
θ_i	Ensemble des paramètres de la variable X_i.
θ_{ijk}	Paramètres de la variable X_i lorsque $\Pi_i = \pi_{ij}$ et $X_i = x_i^k$.
$D_i^{(l)}$	Valeur prise par la variable X_i dans le l^e cas de la base D.
Π_i	Ensemble des sommets parents du sommet X_i dans un graphe orienté $\mathcal{G}$.
π_{ij}	j^e instanciation de Π_i.
q_i	Nombre d'instanciations distinctes de Π_i, $q_i = \prod r_h$, $h : X_h \in \Pi_i$.
N_{ijk}	Nombre de cas, dans la base D, où $X_i = x_i^k$ alors que $\Pi_i = \pi_{ij}$.
$\mathcal{G}_i \equiv \mathcal{G}_j$	Relation d'équivalence au sens de Markov entre les structures $\mathcal{G}_i$ et $\mathcal{G}_j$
Notations probabilistes	
P	Mesure de probabilités
$(. \perp\!\!\!\perp .)$	Relation d'indépendance marginale.
$(. \perp\!\!\!\perp .\|.)$	Relation d'indépendance conditionnelle.
Notations graphiques	
$\mathcal{G}$	Graphe orienté sans circuit constitué de n sommets $\{X_1, X_2, \dots, X_n\}$.
$\mathcal{V}$	Ensemble des n sommets $\{X_1, X_2, \dots, X_n\}$ d'un graphe $\mathcal{G}$.
$\mathcal{E}$	Ensemble des arcs d'un graphe orienté $\mathcal{G}$.
$Adj_{\mathcal{G}}(X)$	Ensemble des sommets de $\mathcal{G}$ directement reliés au sommet X.
$X - Y$	Les sommets X et Y sont reliés par une arête.
$X \rightarrow Y$	Les sommets X et Y sont reliés par un arc allant de X vers Y.
$(. \perp_d .\|.)$	Relation de d-séparation dans un graphe $\mathcal{G}$.
$SepSet_{\mathcal{G}}(X, Y)$	Ensemble de sommets d-séparant les sommets X et Y dans le graphe $\mathcal{G}$

Abréviations

Les abréviations employées :

Notation	Définition
GOSC	Graphe Orienté Sans Circuit.
GPOSC	Graphe Partiellement Orienté Sans Circuit.
GE	Graphe Essentiel.
PAG	*Partial Directed Acyclic Graph* : graphe sans circuit partiellement orienté.
PAG	*Partial Ancestral Graph* : Graphe complet partiellement ancestral.
EDA	*Estimation of Distribution Algorithm* : algorithme à estimation de densité.
EP	*Evolution Programming* : programmation évolutionnaire.
ES	*Evolution Strategies* : stratégies d'évolution.
GP	*Genetic Programming* : programmation génétique.

Première partie

État de l’art

Chapitre 2

Réseaux bayésiens

À travers ce chapitre, nous allons présenter ce que sont les réseaux bayésiens, leur utilité et quelles sont les propriétés fondamentales qui en font une modélisation particulièrement avantageuse. Le sujet étant très étendu, nous ne saurions le traiter exhaustivement. Nous pouvons néanmoins recommander plusieurs ouvrages au lecteur souhaitant approfondir le sujet. Bien entendu, l'ouvrage de référence demeure celui de J. Pearl [Pearl, 1988] qui est à l'origine du formalisme tel que nous le connaissons aujourd'hui. [Charniak, 1991] ou le livre de P. Naïm *et al* [Naïm et al., 2004] fournissent tous deux une très bonne introduction au sujet. Enfin d'autres ouvrages reconnus traitent des réseaux bayésiens ou des modèles graphiques en général : [Lauritzen, 1998, Jordan, 2004, Pearl, 2000].

2.1 Introduction

Dans le cadre de la théorie des probabilités, il est fréquent de chercher à modéliser une distribution de probabilités jointe P sur un domaine de variables aléatoires $\mathcal{U} = \{X_1, X_2, \ldots X_n\}$. La connaissance de cette distribution de probabilités permet de calculer la probabilité de chaque combinaison d'instances distinctes des variables de $\mathcal{U}$. Ceci permettant, étant donné la connaissance des valeurs de certaines variables, de pouvoir calculer la probabilité de différents événements dont les valeurs sont inconnues.

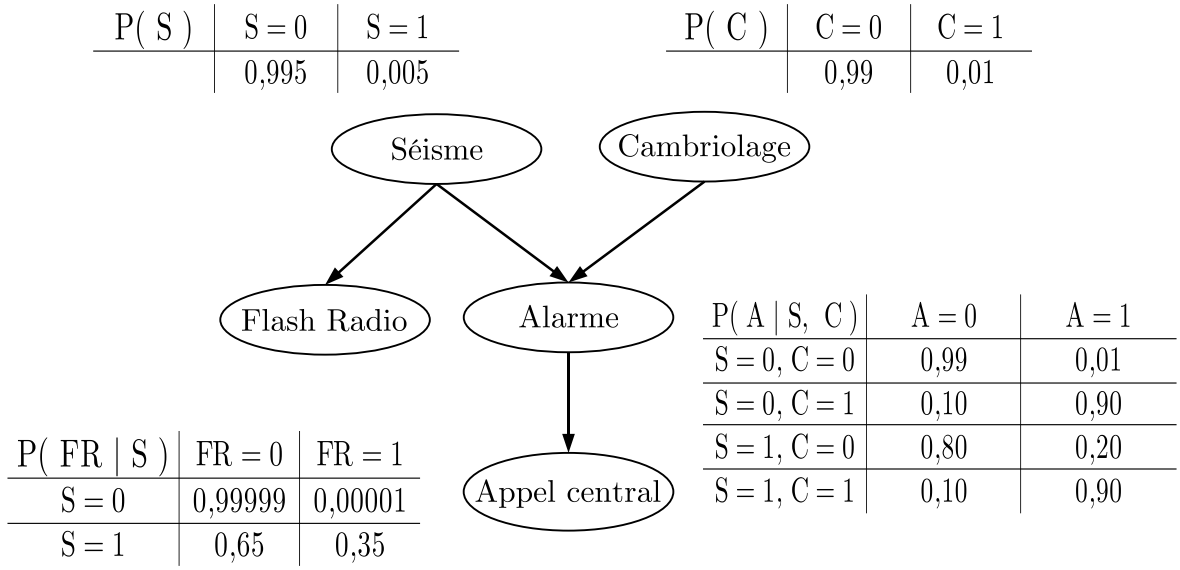
Les réseaux bayésiens font partie d'une branche spécifique de la famille des modèles graphiques probabilistes et se présentent sous la forme d'un graphe orienté sans circuit (ou *GOSC*) symbolisant les différentes dépendances existant entre les variables représentées.

Un réseau bayésien est défini par les éléments suivants :
- un graphe orienté sans circuit dont les sommets représentent des variables aléatoires d'un domaine ;
- les arcs du graphe indiquent des dépendances conditionnelles entre les sommets ;
- des probabilités conditionnelles permettent de quantifier les dépendances.

Un exemple de réseau bayésien est donné dans la figure 2.1. Il s'agit d'un réseau décrivant les relations conditionnelles existant entre :

- la survenue éventuelle d'un séisme ;
- la diffusion d'un flash radio annonçant un séisme ;
- le cambriolage d'un édifice ;
- le déclenchement de l'alarme de cet édifice, suite à un séisme ou un cambriolage ;
- le fait que le central de la compagnie de sécurité appelle les lieux, ou non suivant l'état de l'alarme.

À chaque sommet du graphe est associée une table de probabilités permettant de déterminer la probabilité avec laquelle la variable associée peut prendre une valeur particulière étant donné celles prises par ses parents (s'ils existent).



P(S)	S = 0	S = 1
	0,995	0,005

P(C)	C = 0	C = 1
	0,99	0,01

P(FR | S)	FR = 0	FR = 1
S = 0	0,99999	0,00001
S = 1	0,65	0,35

P(A | S, C)	A = 0	A = 1
S = 0, C = 0	0,99	0,01
S = 0, C = 1	0,10	0,90
S = 1, C = 0	0,80	0,20
S = 1, C = 1	0,10	0,90

P(AC | A)	AC = 0	AC = 1
A = 0	0,95	0,05
A = 1	0,3	0,70

Figure 2.1 – Exemple de réseau bayésien.

Nous remarquons immédiatement certaines indépendances conditionnelles : être cambriolé ou non ne dépend pas de la survenue d'un tremblement de terre (cela pourrait être sujet à débat mais nous admettons cette indépendance par souci de simplicité).

Un des principaux avantages du formalisme des réseaux bayésiens est de permettre une lecture facilitée des indépendances conditionnelles au sein de la distribution de probabilités modélisée. La détermination de ces indépendances permet par la suite la simplification des calculs, souvent fastidieux, nécessaires au calcul de la probabilité d'une instanciation du domaine (*i.e.* la probabilité jointe de ce dernier).

2.2 Définition

Un modèle graphique probabiliste permet de représenter un ensemble de relations conditionnelles au sein d'un domaine $\mathcal{U} = \{X_1, X_2, \ldots X_n\}$ de variables aléatoires ayant chacune leur propre domaine de définition.

Une valeur d'intérêt est la distribution de probabilités jointe spécifiant la probabilité d'apparition des différentes combinaisons de valeurs de variables du domaine. Cette distribution, une fois connue, permet d'estimer la probabilité des valeurs d'une ou plusieurs variables en connaissant les valeurs prises par les autres variables du domaine.

Définition 1 *Un réseau bayésien $\mathcal{B}$ est défini à la fois qualitativement et quantitativement par un couple $(\mathcal{G}, \Theta)$:*

- *$\mathcal{G}$ est un GOSC dont les sommets correspondent aux variables $(X_1, X_2, \ldots X_n)$ de l'ensemble $\mathcal{U}$. Les arcs orientés de $\mathcal{G}$ représentant des dépendances directes entre ces variables.*
- *Θ est l'ensemble des paramètres du réseau. Θ contient les paramètres $\theta_{i,j,k} = P(X_i = x_i^k | \Pi_i = \pi_{ij})$, $i \in 1 \ldots n$ pour chaque valeur x_i^k pouvant être prise par X_i et chaque configuration π_{ij} de Π_i, ensemble des sommets parents de X_i dans $\mathcal{G}$.*

Il est à noter que l'adjectif *bayésien* peut s'avérer trompeur. D'un point de vue bayésien, les probabilités d'occurrence d'un événement, conditionnellement ou non à un autre, sont quantifiées de manière subjective en définissant un *a priori* sur leur distribution. Une approche fréquentiste, quand à elle, repose sur l'observation de séries d'expériences (pour plus de détails, se référer au chapitre 3). S'il est évidemment possible d'employer indifféremment les réseaux bayésiens dans l'un ou l'autre de ces cadres, le terme *bayésien* est employé dans la dénomination du modèle afin de souligner la prépondérance des axiomes relatifs aux probabilités conditionnelles dans la définition et l'usage de ces modèles.

2.3 Propriétés des réseaux bayésiens

L'emploi des réseaux bayésiens permet d'associer la théorie des probabilités à la théorie des graphes. Il convient dès lors de pouvoir lier les propriétés graphiques de la structure $\mathcal{G}$ d'un réseau bayésien $\mathcal{B}$ avec les propriétés de la distribution de probabilités modélisée. L'ensemble des (in)dépendances conditionnelles du domaine peut être déterminé graphiquement à partir d'un ensemble d'axiomes [Pearl, 1997] et d'hypothèses.

La lecture des indépendances conditionnelles sur un graphe est intimement liée à la notion de *séparation*.

La séparation est un critère permettant de statuer si deux sous ensembles de sommets disjoints d'un graphe sont ou non séparés l'un de l'autre étant donné un troisième sous ensemble disjoint.

La séparation est définie différemment selon le type de graphe auquel on s'intéresse (orienté ou non-orienté notamment). Ici, nous nous limitons à la définition de la séparation dans le cadre des graphes orientés.

2.3.1 Condition locale de Markov

Définition 2 *Étant donné un réseau bayésien $\mathcal{B} = \{\mathcal{G}, \Theta\}$, toute variable X_i de $\mathcal{B}$ est indépendante de l'ensemble $Nd(X_i)/\Pi_i$, formé de l'ensemble de ses non-descendants dans $\mathcal{G}$ privés de ses parents, étant donné ces derniers, i.e. :*

$$\forall X_i \in \mathcal{G}, X_i \perp\!\!\!\perp \{Nd(X_i)/\Pi_i\} | \Pi_i$$

Un descendant d'une variable X_i dans un graphe $\mathcal{G}$ est défini comme étant un sommet atteignable depuis X_i par un chemin orienté.

Reprenons l'exemple de la figure 2.1. La condition locale de Markov appliquée ici permet, entre autres, d'affirmer que *Appel Central* est indépendant de *Flash Radio* (qui n'est ni un parent, ni un descendant) connaissant *Séisme* (qui est un parent).

La condition locale de Markov permet donc de détecter un ensemble minimal d'indépendances probabilistes entre les sommets et leurs non-descendants, impliquant entre autres que deux sommets non adjacents X_i et X_j de $\mathcal{G}$ sont conditionnellement indépendants étant donné un troisième sous-ensemble, contenu dans $\mathcal{U}/\{X_i, X_j\}$.

C'est l'application de la condition locale de Markov qui nous permet d'écrire la probabilité jointe des variables du domaine sous une forme factorisée :

$$P(X_1, X_2, \ldots X_n) = \prod_{i=1}^{n} P(X_i | \Pi_i) \tag{2.1}$$

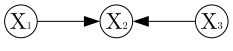
2.3.2 d-séparation

Si, dans un *GOSC*, les relations entre les paires de variables sont binaires (reliées ou non), la détermination d'une indépendance conditionnelle implique généralement trois sous ensembles de variables.

La d-séparation est un critère permettant de déterminer les indépendances conditionnelles modélisées par un *GOSC*. Simplement, il s'agit de déterminer si un sous-ensemble X de variables du domaine est conditionnellement indépendant d'un sous-ensemble Y étant donné un sous-ensemble Z.

S'il paraît évident que nous faisons alors la corrélation entre la connexité et la dépendance conditionnelle, la direction des arcs impliqués entre aussi en jeu (le *d* de d-séparation provenant de *directional*) car nous définissons la notion de chemin connecteur (et inversement, de chemin bloquant). Nous allons introduire progressivement, en les illustrant, les diverses notions nécessaires à la définition de la d-séparation.

Nous emploierons par la suite le terme de *convergence* pour désigner une configuration particulière au sein du graphe.

Définition 3 (V-structure) *Dans un graphe $\mathcal{G}$, on appelle convergence (ou V-structure), tout triplet $\{X_1, X_2, X_3\}$ de sommets tel que* 

Définition 4 (Chemin) *Dans un graphe $\mathcal{G}$, un chemin entre deux sommets A et B de $\mathcal{G}$ désigne une série d'arcs consécutifs reliant A à B, quelle que soit leur orientation.*

Définition 5 (Chemin bloquant) *Dans un graphe $\mathcal{G}$, un chemin entre deux sommets A et B de $\mathcal{G}$ est dit bloquant s'il comporte au moins une convergence de la forme $X_1 \rightarrow X_2 \leftarrow X_3$ telle que X_2 ne soit pas instanciée.*

La dernière précision, concernant l'instanciation au sein des convergences, sera expliquée plus loin dans cette section.

Notre définition de la d-séparation repose sur celle de sa contraposée : la d-connexion.

Séparation inconditionnelle

Soient deux variables X_1 et X_2. X_1 et X_2 sont d-connectées s'il existe un chemin non-bloquant entre X_1 et X_2.

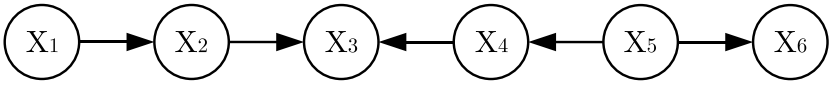


Figure 2.2 – Illustration de la séparation inconditionnelle.

Dans la figure 2.2, les sommets X_1 et X_3 sont d-connectés de même que les sommets X_4 et X_6. En revanche, on distingue une convergence (non instanciée) sur le sommet X_3. Celle-ci implique, entre autres, que les sommets X_1 et X_6 ne sont pas d-connectés (et par conséquent, sont d-séparés).

Blocage conditionnel

Considérons un sous ensemble Z de variables aléatoires d'un domaine $\mathcal{U}$. Si les valeurs prises par ces variables sont connues, la distribution de probabilités, conditionnellement à ce sous-ensemble, est modifiée qualitativement. Il convient alors de définir la d-connexion par rapport à un ensemble de conditions pouvant bloquer cette connexion.

Deux sommets X_1 et X_2 sont d-connectés conditionnellement à un sous-ensemble Z de sommets si il existe un chemin sans convergence reliant X_1 et X_2 et ne passant par aucune des variables de Z.

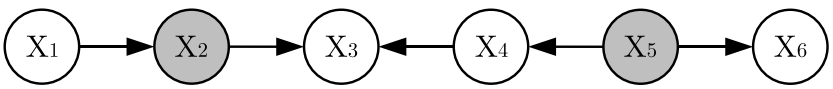


Figure 2.3 – Illustration d'un blocage conditionnel.

En considérant, dans la figure 2.3 que les sommets grisés appartiennent au sous-ensemble Z : X_1 et X_6 sont toujours d-séparés mais, de plus, X_1 et X_3 sont d-séparés par Z (en raison de X_2) ainsi que X_4 et X_6.

Conditionnement sur les convergences

Si on observe un événement ayant deux causes distinctes et originellement indépendantes, ces causes deviennent dépendantes. Un exemple intuitif permettant de comprendre ce principe est celui du lancer de deux pièces. Les variables X_1 et X_2 représentent chacune le résultat du lancer d'une des deux pièces et la variable aléatoire X_3 vaut 1 si les deux lancers ont eu le même résultat et 0 sinon. Il est alors évident que la connaissance de X_3 crée une dépendance entre X_1 et X_2.

Ce résultat, aussi connu sous le nom de paradoxe de Berkson, implique un enrichissement des deux points précédents et plus exactement dans le cas des sommets situés sur des convergences (deux causes communes) et leurs descendants.

Si un sommet convergent se trouvant sur le chemin appartient à l'ensemble conditionnant Z ou à un de ses descendants dans Z, il n'est plus un facteur bloquant de d-connexion.

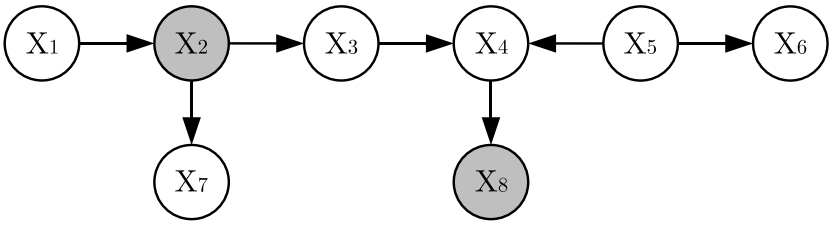


Figure 2.4 – Illustration de d-connexions et d-séparations avec conditionnement sur les variables.

Sur la figure 2.4, X_1 et X_3 sont d-séparés par $X_2 \in Z$ de même que X_1 et X_7. Mais X_3 et X_6 sont d-connectés puisque X_4 a son unique descendant dans Z.

Définition 6 (Chemins bloqués, actifs) *Soit $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, un graphe orienté sans circuit. Soit A, B et C trois sous ensembles disjoints de $\mathcal{V}$. Soit ρ un chemin reliant un sommet de A à un sommet de B. Le chemin ρ est dit bloqué par l'ensemble C si une des deux conditions suivantes est remplie :*

- *Le chemin ρ converge en un sommet X_i et ni X_i ni aucun de ses descendants ne sont dans C*
- *Le chemin passe par un sommet $X_i \in C$ en lequel il n'y a pas de convergence*

Si aucune de ces conditions n'est remplie, on dit alors que le chemin ρ est actif.

Cette définition permet alors de définir précisément le critère de d-séparation :

Définition 7 (d-séparation) *Soit $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, un graphe orienté sans circuit. Soit A, B et C trois sous ensembles disjoints de $\mathcal{V}$. A et B sont d-séparés par C dans $\mathcal{G}$ (noté $(A \perp_{\mathcal{G}} B|C)$) si et seulement si tous les chemins reliant un sommet de A à un sommet de B sont bloqués par C.*

Par la suite, nous emploierons la notation $A \perp_{\mathcal{G}} B$ pour indiquer que A et B sont d-séparés dans $\mathcal{G}$.

2.3.3 Cartes d'indépendance

Les règles de la d-séparation que nous venons de voir nous ont permis de déterminer un ensemble de relations ternaires non-explicites faisant intervenir des sous ensembles disjoints de sommets.

Une fois les relations de séparation au sein du graphe $\mathcal{G}$ détectées, nous allons chercher à qualifier $\mathcal{G}$ par rapport à la distribution de probabilités P du domaine que nous caractérisons par l'ensemble des relations d'indépendances conditionnelles qu'elle implique.

Définition 8 (Carte d'indépendance) *Soit P une distribution de probabilités sur un ensemble de variables aléatoires $\mathcal{U}$; $\mathcal{G}$, un GOSC composé sur $\mathcal{U}$ et $X, Y, Z \subseteq \mathcal{U}$, et $(. \perp\!\!\!\perp_P .|.)$ une relation d'indépendance conditionnelle vérifiée par P.*

– *$\mathcal{G}$ est une carte d'indépendance (ou I-map) de P s'il vérifie :*

$$X \perp_{\mathcal{G}} Y|Z \Rightarrow X \perp\!\!\!\perp_P Y|Z$$

– *$\mathcal{G}$ est une carte de dépendance (ou D-map) de P s'il vérifie :*

$$X \perp\!\!\!\perp_P Y|Z \Rightarrow X \perp_{\mathcal{G}} Y|Z$$

– *$\mathcal{G}$ est une carte parfaite (ou P-map) de P s'il vérifie :*

$$X \perp_{\mathcal{G}} Y|Z \Leftrightarrow X \perp\!\!\!\perp_P Y|Z$$

A noter que dans le cas d'une I-map, nous admettons la possibilité qu'il existe des indépendances conditionnelles de P qui ne sont pas représentées dans $\mathcal{G}$. Un graphe entièrement connecté est alors une I-map de toutes les lois de probabilités sur $\mathcal{U}$.

Cette notion peut être illustrée très simplement par l'exemple suivant : considérons la distribution de probabilités P définie sur deux variables X et Y et décrite dans la figure 2.5. Les variables X et Y sont indépendantes selon la distribution P.

X	Y	$P(X,Y)$
0	0	0,42
0	1	0,28
1	0	0,18
1	1	0,12

$$\forall x, y \in \{0,1\}^2 :$$
$$P(X = x, Y = y) = P(X = x) \times P(Y = y)$$
$$\Leftrightarrow X \perp\!\!\!\perp_P Y$$

Figure 2.5 – Distribution de probabilités P définie sur deux variables X et Y.

Cependant, les graphes $\mathcal{G}_1$ et $\mathcal{G}_2$ de la figure 2.6 sont tous les deux des I-maps des distributions de probabilités associées : X et Y sont bien indépendantes selon $\mathcal{G}_1$ et l'ensemble des indépendances conditionnelles représentées par $\mathcal{G}_2$ est l'ensemble vide, ce qui respecte bien la définition d'un I-map.

Par définition [Verma et Pearl, 1990], un réseau bayésien est toujours une I-map de la distribution qu'il représente. L'inconvénient de cette définition est qu'un *GOSC* entièrement connecté est lui aussi une I-map de la distribution encodée (puisqu'il n'encode aucune indépendance).

Figure 2.6 – Exemple de cartes d'indépendances.

Définition 9 (Carte d'indépendances minimale) *$\mathcal{G}$ est une carte d'indépendances minimale de la distribution de probabilités P si aucun graphe partiel $\mathcal{G}'$ de $\mathcal{G}$ n'est une carte d'indépendances de P.*

Concrètement, cette définition signifie qu'aucun arc de $\mathcal{G}$ ne peut être retiré sans violer la propriété des cartes d'indépendances.

Théorème 1 *Un graphe $\mathcal{G}$ est une I-map de la distribution de probabilités P si et seulement si P peut se factoriser selon $\mathcal{G}$:*

$$P(X_1, X_2, \ldots X_n) = \prod_{i=1}^{n} P(X_i|\Pi_i), \forall X_i \in \mathcal{U}$$

2.3.4 Factorisation de la probabilité jointe

Les différentes propriétés précédemment énoncées ont pour finalité de permettre l'exploitation de l'interface graphique du réseau bayésien afin de simplifier le calcul de la probabilité jointe.

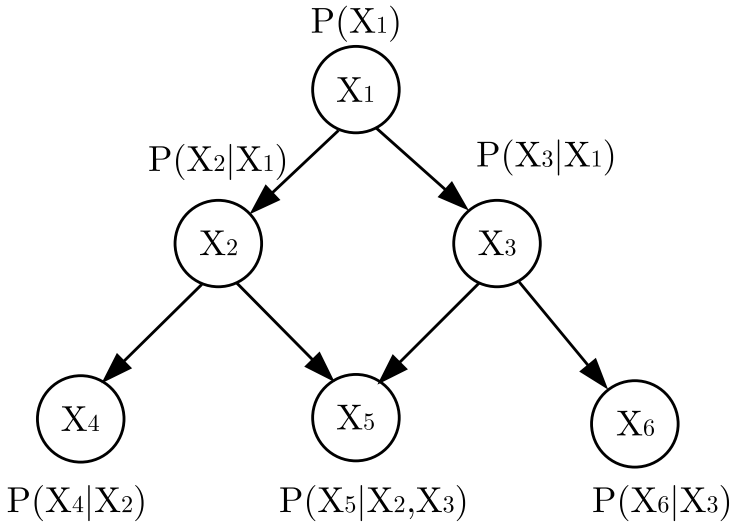


Figure 2.7 – Exemple de réseau bayésien.

Soit le réseau bayésien de la figure 2.7. Ce réseau est constitué de 6 variables binaires ; le calcul de la probabilité jointe de ces 6 variables requerrait $2^6 - 1 = 63$ paramètres indépendants.

Nous nous sommes restreints ici à un exemple simple, il va de soi que dans le cadre d'une modélisation plus réaliste, à la fois le nombre de variables et les cardinalités de celles-ci seraient beaucoup plus élevées.

L'équation du théorème 1, appliquée au réseau de la figure 2.7 nous donne alors la décomposition suivante :

$$P(X_1, X_2, \ldots, X_6) = P(X_1) \times P(X_2|X_1) \times P(X_3|X_1) \times P(X_4|X_2) \times P(X_5|X_2, X_3) \times P(X_6|X_3)$$

Ici, le calcul de la probabilité jointe ne nécessite plus que le calcul de $1 + 2 + 2 + 2 + 4 + 2 = 13$ entrées indépendantes. L'économie en calculs devient bien sûr d'autant plus impressionnante que le nombre de variables du réseau concerné est grand (et le graphe parcimonieux).

2.4 Causalité dans les réseaux bayésiens

Jusqu'à présent, nous avons défini et employé les réseaux bayésiens conjointement au terme de causalité en raison de l'importance de l'orientation de la structure du modèle dans son utilisation pratique. Or, il est important de pouvoir distinguer un modèle *statistique* et un modèle *causal*.

Les réseaux bayésiens peuvent être de deux types : causaux ou non-causaux. Un réseau bayésien causal modélise expressément un ensemble de relations de type cause à effet : chaque sommet non-racine du graphe est la conséquence directe de ses parents dans le graphe. Un réseau non causal, en revanche, modélise des relations de dépendance *probabilistes* entre les variables : un arc allant d'un sommet X vers un sommet Y n'implique pas une relation de causalité.

Jusqu'ici nous avons établi une définition formelle des réseaux bayésiens sans nous intéresser explicitement aux méthodes d'apprentissage de ces derniers. Or, nous verrons par la suite que les méthodes d'apprentissage usuelles, employant une base d'apprentissage constituée d'exemples d'instances du domaine, ne permettent d'apprendre la structure d'un réseau bayésien qu'à sa classe d'équivalence au sens de Markov près (cf. section 4.4.4) ; dans le cas où le graphe est causalement suffisant (cf. section 4.1.1), seuls les arcs orientés au sein du graphe partiellement orienté servant de représentant à la classe d'équivalence du graphe représentent des causalités effectivement déterminées par l'information contenue dans la base d'apprentissage.

La détermination des différents liens de causalité dans un graphe non-causal peut alors se faire de deux manières. Soit par l'intervention d'un expert à qui incombe la tâche de représenter les différents liens (et donc orientation des arcs dans la structure), soit par l'observation des conséquences qu'ont des interventions locales en certaines variables sur le domaine modélisé.

L'intérêt principal de la causalité entre dans le cadre de *l'inférence causale* qui a pour objectif de pouvoir mesurer l'effet d'une intervention sur une ou plusieurs variables sur la probabilité d'un ensemble d'autres variables. L'inférence causale et la notion de causalité en général sont toujours sujets à discussion aujourd'hui, tant sur le plan mathématique que sur le plan philosophique. Le lecteur pourra néanmoins se reporter à la lecture de [Spirtes et al., 1999, Pearl, 2000, Meganck et al., 2006a, Meganck et al., 2006b, Murphy, 2003] pour plus de détails sur les réseaux bayésiens causaux ainsi que sur leur apprentissage.

Dans le cadre de nos travaux, nous nous restreignons à l'apprentissage des structures à partir de bases d'apprentissage statiques et donc à l'apprentissage de réseaux bayésiens non causaux.

2.5 Réseaux bayésiens à densités de probabilités continues

Les réseaux bayésiens, tels que présentés dans ce travail de thèse, comportent des variables prenant leurs valeurs dans des espaces discrets. Bien qu'il soit répandu de travailler sur de tels modèles, essentiellement pour des raisons pratiques, il est tout à fait possible d'employer des réseaux bayésiens dans le cas où les variables modélisées sont continues. Ainsi, [Lauritzen et Wermuth, 1989] ont proposé des réseaux bayésiens dont les variables présentent une densité de probabilités correspondant à une distribution gaussienne. D'autres modélisations permettent de généraliser la densité modélisée en l'approximant par un mélange de gaussiennes [Lerner et al., 2001] ou encore des densités exponentielles tronquées [Cobb et Shenoy, 2006].

2.6 Inférence probabiliste

Les réseaux bayésiens permettent la décomposition de la probabilité jointe des variables du domaine qu'ils modélisent. Cette dernière propriété permet de pouvoir déduire la valeur prise par une ou plusieurs variables du domaine à partir de l'observation d'autres variables [Heckerman et al., 1995b].

Commençons par définir (O, H), une partition de l'ensemble des variables $\mathcal{X}$. H représente l'espace des hypothèses. Une hypothèse h représente une certaine instanciation des variables de H. O représente, comparativement, les instanciations des variables de O, valeurs connues au moment de l'inférence.

L'inférence probabiliste revient couramment à calculer l'un des deux termes suivants :

Probabilités marginales

$$P(o) = \sum_{h' \in H} P(o, h')$$

Probabilités *a priori* maximales

$$P_{max}(o) = max_h P(o, h)$$

Il se peut que l'on cherche aussi, pour une instance de h, la valeur de sa probabilité conditionnelle selon o :

$$P(h|o) = \frac{P(o, h)}{\sum_{h' \in H} P(o, h')}$$

Pour résumer, l'inférence probabiliste revient à calculer, étant donné le modèle et un ensemble d'observations (on parle, dans la littérature anglo-saxonne, de "preuve" ou *evidence*), la probabilité pour une instanciation supposée des variables demeurant ou bien quelle instanciation de celles-ci est la plus probable.

Les méthodes d'inférence peuvent se répartir en deux groupes : les méthodes exactes et les méthodes approchées.

Parmi les méthodes exactes, l'algorithme du *message passing* (passage de messages) [Pearl, 1988] – restreint aux graphes formant un arbre – ou encore du *junction tree* (arbre de jonction) [Jensen et al., 1990] figurent parmi les plus usités. Ces algorithmes sont expliqués en détail dans [Pearl, 1997] et [Huang et Darwiche, 1996]. D'autres possibilités sont l'élimination de variables [Dechter, 1997] ou les méthodes symboliques permettant de limiter les calculs dans les cas les plus complexes [Li et D'Ambrosio, 1994].

Les méthodes exactes cherchent à limiter la quantité de calculs nécessaires en traitant les variables de manière locale ; en les regroupant en cliques, par exemple. Néanmoins, cette simplification rencontre vite ses limites dans le cas de réseaux trop complexes pour être traités de la sorte. On peut alors décider de continuer à traiter le problème de manière exacte mais en ne travaillant que sur une partie du réseau. Parmi les méthodes approchées, les plus connues sont celles basées sur le principe de *Monte Carlo Markov Chain* ou MCMC [MacKay, 1998]. Les méthodes d'échantillonage de Gibbs ou de Metropolis-Hastings [Lauritzen, 1998] peuvent ainsi être appliquées aux réseaux bayésiens. L'approximation peut aussi s'opérer en se limitant à un sous ensemble de variables [Draper et Hanks, 1994] ou bien en évaluant les sommations impliquées durant une inférence de type exact [D'Ambrosio, 1993]. On peut, de même, limiter le réseau sur lequel a lieu l'inférence en ignorant les dépendances les plus faibles en son sein [Kjærulff, 1994].

Enfin, les méthodes dites *variationnelles* cherchent, quant à elles, à déterminer le maximum de vraisemblance en approximant la probabilité *a posteriori* [Jaakkola et Jordan, 1999, Beal, 2003].

L'inférence, exacte ou approchée, a été montrée comme étant un problème NP-difficile [Cooper, 1987, Dagum et Luby, 1993] et le sujet est présenté plus en détail dans le livre de F. Jensen [Jensen, 1996]. De même, l'ouvrage de M. Jordan [Jordan, 1998] regroupe une série de tutoriaux et d'articles sur les réseaux bayésiens mais aussi sur les modèles graphiques en général.

2.7 Exemples d'application de réseaux bayésiens

Les applications des réseaux bayésiens s'attachent essentiellement à la prédiction, au diagnostic et à l'assistance à la décision :

Filtrage du pourriel concept initialisé par [Sahami et al., 1998]. L'utilisation des réseaux bayésiens pour le filtrage du courrier indésirable s'est popularisée et figure parmi les applications les plus réussies et populaires des réseaux bayésiens.

Assistance aux handicapés PAM-AID [Lacey et MacNamara, 2000] est un système d'assistance au déplacement en intérieur à destination des personnes à mobilité réduite. Concrétement, le système consiste en un déambulateur motorisé pouvant détecter les obstacles (murs, objets,...) lors du déplacement.

L'assistance au pilotage C'est le cas pour la NASA avec le système VISA servant au diagnostic des systèmes de propulsion.

Décisions tactiques SAIP (*Semi-Automated IMINT Processing*) [Fennell et Wishner, 1998] est un programme du DARPA (*Defense Advanced Research Projects Agency*) visant à fournir au commandement militaire une information tactique à partir d'images haute définition. Des systèmes tels que les réseaux bayésiens interviennent dans le pré-traitement des images afin de déterminer les priorités tactiques des éléments sur le terrain.

Aide à l'interaction Plus récent, le programme Genoa II [Allanach et al., 2004], issu lui aussi de la recherche au DARPA, a pour objectif l'amélioration des interactions homme-machine dans le cadre de la lutte anti-terroriste et emploie à ces fins divers outils bioinformatiques ainsi que la modélisation bayésienne.

Évaluation du risque EDF emploie les réseaux bayésiens afin de prévoir les risques liés à la disponibilité des sources froides (*i.e.* le débit fluvial) pour les centrales nucléaires situées le long de la Loire.

Études de marché Les réseaux bayésiens, en conjonction avec des études expertes, peuvent permettre de mieux cerner les besoins et impératifs commerciaux d'une entreprise en précisant, par exemple, le cœur de cible d'une agence bancaire [Jaronski et al., 2001].

Les exemples d'application sont très nombreux et l'on ne saurait en faire une liste exhaustive. Mais l'intérêt grandissant, depuis le milieu des années quatre-vingt dix, dont ont fait preuve les industriels pour les modèles bayésiens ne fait que croître en particulier grâce à la généralisation de processus d'interaction entre l'homme et la machine pour accélérer les prises de décision.

Parmi les avantages proposés par les réseaux bayésiens, nous pouvons aussi mentionner leur capacité, en conjugaison avec les méthodes statistiques dites bayésiennes (c'est-à-dire prenant en compte un *a priori* sur la distribution de probabilités modélisée) à conjuguer la connaissance extraite de la base de connaissance avec une connaissance préalable du domaine. Cette connaissance, subjective, est fréquemment le produit de l'avis d'un expert humain sur le sujet. Cette propriété est appréciable lorsque l'on sait que dans l'application pratique, l'acquisition de données est non seulement coûteuse en moyens et en temps mais, hélas, débouche souvent sur une base de connaissance de taille réduite.

Nous verrons de plus, dans les chapitres suivants, que l'apprentissage des réseaux bayésiens peut aussi s'effectuer à partir de bases de données incomplètes (*i.e.* bases pour lesquelles les valeurs prises par certaines variables du domaine sont inconnues pour certaines instances).Cette possibilité est particulièrement intéressante quand le processus de fouille de données ne peut systématiquement retourner l'ensemble des valeurs prises par les différentes composantes du modèle (à cause de capteurs défectueux, par exemple).

2.8 Conclusion

Nous avons jusqu'ici abordé les fondements théoriques ainsi que les applications des réseaux bayésiens. Dans la suite, nous nous intéressons à l'apprentissage. L'apprentissage d'un réseau bayésien peut se décomposer en deux phases. Dans un premier temps, la structure du réseau est déterminée, soit par un expert, soit de manière automatique à partir d'une base de cas issus du domaine modélisé (le plus souvent). Enfin, les paramètres Θ du réseau sont à leur tour déterminés, ici aussi par un expert ou bien par le biais d'un algorithme.

Nos travaux concernent l'apprentissage de la structure, c'est cet aspect de l'apprentissage que nous allons le plus développer. Cependant, il nous paraît indispensable de présenter les bases essentielles de l'apprentissage des paramètres sans lesquels le réseau ne saurait être entièrement déterminé.

Chapitre 3

Apprentissage des paramètres

La démarche usuelle, lors de l'élaboration automatique d'un réseau bayésien, consiste d'abord par inférer la structure du réseau à partir d'une base de données puis à l'apprentissage des paramètres de ce réseau.
Dans ce chapitre, nous allons nous intéresser à l'apprentissage des paramètres.

Nous décrivons dans cette partie les méthodes les plus employées pour cette tâche.

3.1 Généralités

Deux principaux cas de figure peuvent se présenter :
– le cas où l'ensemble des données contenues dans D est observé ;
– celui où certaines valeurs, pour une instanciation $D^{(l)}$ donnée, ne sont pas connues.
Auparavant, plusieurs hypothèses doivent être préalablement faites afin de pouvoir effectuer les calculs nécessaires [Heckerman et Geiger, 1994].

Indépendance des échantillons : les éléments de la base D sont indépendants, identiquement distribués ;

Indépendance des paramètres : pour toute structure $\mathcal{G}$, d'une part les paramètres θ_i associés au nœud X_i sont indépendants des paramètres associés aux autres nœuds et d'autre part, les paramètres θ_{ij} associés à X_i suivant une instanciation π_{ij} des parents de X_i dans $\mathcal{G}$ sont indépendants des paramètres associés aux autres instanciations de Π_i ;

Modularité paramétrique : si un nœud présente les mêmes parents dans deux structures distinctes $\mathcal{G}_1$ et $\mathcal{G}_2$, alors la densité de distribution des paramètres de X_i est la même pour les deux deux modèles définis par $\mathcal{G}_1$ et $\mathcal{G}_2$;

Une autre hypothèse, courante, est de supposer que les paramètres de chaque nœud suivent des densités de probabilités admettant des paramètres de Dirichlet. Les différents paramètres θ_i admettent alors une densité de probabilité exponentielle de Dirichlet d'exposants $\alpha_1, \ldots, \alpha_r$

et leur probabilité est égale à :

$$P(\theta_i|\alpha_1,\ldots,\alpha_n) = \frac{\Gamma\left(\sum_{i=1}^{r_i}\alpha_i\right)}{\prod_{i=1}^{r_i}\Gamma(\alpha_i)} \times \prod_{i=1}^{r}\theta^{\alpha_i-1} \tag{3.1}$$

Ici, Γ représente la fonction Gamma d'Euler : $\Gamma(x+1) = x\Gamma(x)$ et $\Gamma(1) = 1$ dans $\mathbb{R}$.

Cette hypothèse sur la densité de probabilités des paramètres vise à simplifier les calculs des paramètres. La première hypothèse revient en effet à dire que les échantillons de la base D est un échantillon d'une loi multinomiale. Or, la densité de probabilité de Dirichlet est la conjuguée de la loi multinomiale. Ceci permet de conserver à la fois les paramètres et leurs densités dans la même famille de fonctions.

Nous allons voir que dans les deux cas, en présence de données complètes ou incomplètes, nous avons le choix entre un apprentissage uniquement basé sur les informations extraites de la base d'apprentissage du modèle et un apprentissage permettant de mêler ces informations à une éventuelle connaissance *a priori* que nous pourrions avoir sur le domaine.

3.2 Base de données complète

Ici, la base d'apprentissage D ne contient pas de cas où viendrait à manquer une ou plusieurs observations.

L'apprentissage des paramètres d'un réseau bayésien peut ici se faire suivant deux approches :
- L'approche statistique (ou fréquentiste)
- L'approche bayésienne (ou subjective)

Dans les deux cas, l'ensemble Θ des paramètres est estimé à partir de la formule de Bayes :

$$P(\Theta|D) = \frac{P(D|\Theta) \times P(\Theta)}{P(D)}. \tag{3.2}$$

Cette équation peut aussi s'écrire sous la forme :

$$\text{Probabilité a posteriori} = \frac{\text{Vraisemblance} \times \text{Probabilité a priori}}{\text{Vraisemblance marginale}}.$$

La probabilité *a posteriori* de l'ensemble des paramètres Θ connaissant la base de cas D est fonction de la vraisemblance de cet ensemble par rapport à D, de la probabilité *a priori* de Θ (*i.e.* la connaissance *a priori* que nous pouvons avoir sur le domaine) et de la vraisemblance de D.

Pour un ensemble de cas $D^{(l)}, l \in 1,\ldots N$ indépendants et identiquement distribués, la vraisemblance des données D étant donné les paramètres Θ estimés s'écrit :

$$P(D|\Theta) = \prod_{l=1}^{N} p(D^{(l)}|\Theta). \tag{3.3}$$

Enfin, reste la valeur de la probabilité *a priori* $P(\Theta)$ dont le calcul constitue la principale différence entre l'approche statistique et l'approche bayésienne, comme nous allons le voir dans la suite.

3.2.1 Approche fréquentiste

Les méthodes fréquentistes utilisent dès lors différents estimateurs dont le but est de parvenir à déterminer la meilleure approximation de la valeur des différents paramètres du réseau.

Un de ces estimateurs est celui du maximum de vraisemblance. Pour chaque variable X_i, la probabilité d'apparition de l'événement x_i est directement proportionnelle à sa fréquence d'apparition dans la base d'apprentissage.

Soit N_{ijk} le nombre d'occurrences simultanées dans la base de $X_i = x_k$ et $\Pi_i = \pi_{ij}$ où $k \in 1, \dots r_i$ et $j \in 1, \dots q_i$.

La probabilité estimée est alors notée $\hat{P}(X_i = x_k | \Pi_i = \pi_{ij}) = \theta_{ijk}^{MV} = \frac{N_{ijk}}{\sum_k N_{ijk}}$

La log-vraisemblance (le logarithme de la vraisemblance) est souvent employée pour des raisons pratiques (manipulation de valeurs numériques très faibles).

$$LL(\Theta|D) = logP(D|\Theta) = \sum_{l=1}^{N} logP(D^l|\Theta)$$

Nous cherchons alors, pour un paramètre $\theta_{ijk} = P(X_i = x_k | \Pi_i = \pi_{ij})$, la valeur $\hat{\theta}_{ijk}^{MV}$ permettant de maximiser localement la vraisemblance, c'est-à-dire en chaque X_i :

$$\hat{\theta}_{ijk}^{MV} = argMax_{\hat{\theta}_{ijk}} LL(\hat{\theta}_{ijk}|D)$$

Avec N_{ijk}, le nombre d'occurences simultanées dans D de $X_i = x_k$ et $\Pi_i = \pi_{ij}$.

Démonstration 1 ([Naïm et al., 2004]) *La vraisemblance $L(D^{(l)}|\Theta)$ d'une instance $D^{(l)}$ issue de la base d'apprentissage D, où $D^{(l)} = \{X_1^{(l)}, X_2^{(l)}, \dots X_n^{(l)}\}$, en connaissance des paramètres du réseau, s'écrit :*

$$\begin{aligned} L(D^{(l)}, \Theta) &= P(X_1^{(l)}, X_2^{(l)}, \dots X_n^{(l)}) && (3.4) \\ &= \prod_{i=1}^{n} P(X_i^{(l)} | \Pi_i^{(l)}) && (3.5) \\ &= \prod_{i} \theta_{ijk}^{(l)} && (3.6) \end{aligned}$$

où $\theta_{ijk}^{(l)}$ indexe implicitement les valeurs spécifiques prises respectivement par X_i et Π_i pour $D^{(l)}$.

Nous supposons que les exemples de la base sont indépendants et identiquement distribués, ce qui nous permet d'écrire la vraisemblance pour l'ensemble de la base D :

$$L(D, \Theta) = \prod_{i=1}^{n} \prod_{l=1}^{N} \theta_{ijk}^{(l)} = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk}} \quad (3.7)$$

la log-vraisemblance s'écrivant alors :

$$LL(\Theta, D) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} log(\theta_{ijk})$$

La vraisemblance et la log-vraisemblance atteignent leur maximum au même point en lequel s'annule donc la dérivée de $LL(\Theta, D)$.

Nous réécrivons la log-vraisemblance :

$$LL(\Theta, D) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \left(\sum_{k=1}^{r_i - 1} \left(N_{ijk} log(\theta_{ijk}) \right) + N_{i,j,r_i} log(\theta_{i,j,r_i}) \right)$$

$$LL(\Theta, D) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \left(\sum_{k=1}^{r_i - 1} \left(N_{ijk} log(\theta_{ijk}) \right) + N_{i,j,r_i} log\left(1 - \sum_{k=1}^{r_i - 1} \theta_{ijk} \right) \right)$$

Les dérivées partielles par rapport aux différents θ_{ijk} *valent*

$$\frac{\partial LL(\Theta, D)}{\partial \theta_{ijk}} = \frac{N_{ijk}}{\theta_{ijk}} - \frac{N_{i,j,r_i}}{1 - \sum_{k=1}^{r_i - 1} \theta_{ijk}}$$

La dérivée de la log-vraisemblance s'annule donc quand chaque $\hat{\theta}_{ijk}$ *vérifie :*

$$\frac{N_{ijk}}{\hat{\theta}_{ijk}} = \frac{N_{i,j,r_i}}{1 - \sum_{k=1}^{r_i - 1} \hat{\theta}_{ijk}}$$

soit

$$\frac{N_{i,j,1}}{\hat{\theta}_{i,j,1}} = \frac{N_{i,j,1}}{\hat{\theta}_{i,j,1}} = \ldots = \frac{N_{i,j,1}}{\hat{\theta}_{i,j,1}} = \frac{\sum_{k=1}^{r_i} N_{ijk}}{\sum_{k=1}^{r_i} \hat{\theta}_{ijk}} = \sum_{k=1}^{r_i} N_{ijk}$$

Au final, nous obtenons bien

$$\hat{\theta}_{ijk} = \frac{N_{ijk}}{\sum_{k=1}^{r_i} N_{ijk}}, \qquad \forall k \in \{1, \ldots, r_i\}$$

Si l'approche fréquentiste paraît naturelle, elle présente néanmoins un inconvénient majeur dans le cas d'une base d'apprentissage de taille limitée ; si une instanciation particulière des variables du domaine peut exister, avec une probabilité faible mais non nulle, mais qu'elle n'est pas présente dans la base d'apprentissage, alors d'après l'approche fréquentiste la probabilité de cette configuration est nulle.

Or, le fait qu'une instanciation n'ait pas été observée ne signifie pas nécessairement qu'elle ait une probabilité nulle. Afin d'y circonvenir, il serait bon de pouvoir exprimer et quantifier la possibilité de la survenance d'un tel événement. C'est cet objectif que remplit l'approche bayésienne.

3.2.2 Approche bayésienne

Le principe de cette approche revient à traiter le paramètre θ_{ijk} comme une variable aléatoire dotée d'une densité de probabilité sur l'intervalle [0,1].

Si les paramètres θ_i admettent une densité de probabilité exponentielle de Dirichlet (cf. equation 3.1) et que la distribution de D suit une loi multinomiale, nous pouvons exprimer la probabilité *a posteriori* des paramètres Θ :

$$P(\Theta) \propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} (\theta_{ijk})^{\alpha_{ijk}-1} \tag{3.8}$$

Nous savons déjà que :
– la vraisemblance $L(D|\Theta)$ est égale à (équation 3.7) :

$$\prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} (\theta_{ijk})^{N_{ijk}}$$

– d'autre part :

$$P(\Theta|D) = \frac{P(D|\Theta) \times P(\Theta)}{P(D)}$$

Nous pouvons dès lors écrire :

$$P(\Theta|D) \propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} (\theta_{ijk})^{N_{ijk}+\alpha_{ijk}-1} \tag{3.9}$$

De la même manière que dans le cas du maximum de vraisemblance abordé dans l'approche fréquentiste, nous pouvons alors rechercher les paramètres non plus selon le maximum de vraisemblance mais selon le *maximum a posteriori (MAP)*. En effet, dans le cadre de l'approche bayésienne, le fait d'employer des *a priori* sur les paramètres du modèle – par l'emploi de coefficients α_{ijk} – sous entend que les données ont déjà été observées. La détermination des paramètres ne se fait alors plus selon les occurrences des données (par vraisemblance) mais conditionnellement à celles-ci (approche *a posteriori*).

$$\hat{\theta}_{ijk}^{MAP} = \hat{P}(X_i = x_k|\Pi_i = \pi_{ij}) = \frac{N_{ijk} + \alpha_{ijk} - 1}{\sum_k (N_{ijk} + \alpha_{ijk} - 1)} \tag{3.10}$$

Une autre approche consiste à non plus rechercher le maximum *a posteriori* mais son espérance (*EAP*) :

$$\hat{\theta}_{ijk}^{EAP} = \hat{P}(X_i = x_k|\Pi_i = \pi_{ij}) = \frac{N_{ijk} + \alpha_{ijk}}{\sum_k (N_{ijk} + \alpha_{ijk})} \tag{3.11}$$

La différence entre ces deux dernières approches consiste essentiellement à définir si l'on souhaite procéder à la *sélection d'un modèle* – auquel cas on cherche le modèle maximisant la probabilité *a posteriori* – ou bien estimer un modèle le plus informatif possible quant aux différentes hypothèses représentées au sein d'une quantité de données limitée – si on désire alors un modèle *prédictif* –.

3.2.3 Différences

Les partisans de l'approche fréquentiste reprochent généralement à la philosophie bayésienne d'attribuer des estimations $P(\theta|D)$ différentes suivant des probabilités *a priori* $P(\theta)$ différentes, introduisant par là une subjectivité forte. L'approche fréquentiste considère que l'ensemble Θ des paramètres a une valeur fixe et ne suit pas une distribution de probabilité.

Les méthodes d'apprentissage des paramètres que nous venons de décrire ne sont valables que si l'ensemble des valeurs de la base de données D est observable. Dans le cas contraire, il est nécessaire de faire appel à des méthodes permettant d'estimer les valeurs des observations manquantes. Ce sont ces méthodes que nous allons évoquer dans la section suivante.

3.3 Base de données incomplète

Il peut arriver qu'au sein de la base d'apprentissage du modèle, certaines valeurs soient manquantes. Cette situation peut arriver dans le cas où, par exemple, un capteur de données est tombé en panne ou encore lorsque le relevé de valeurs s'avère trop coûteux pour être systématiquement appliqué.

Les approches vues jusqu'ici ne sont plus, alors, applicables directement (à moins de ne considérer pour l'apprentissage que les instances complètes de la base). Si la solution la plus simple consiste à ignorer les instances incomplètes de la base de données pour l'apprentissage, il est plus courant d'employer une méthode revenant à estimer les données manquantes à partir des données connues. Ce principe est fondé sur celui de l'algorithme EM (*Expectation Maximisation* ou *Espérance Maximisation*) proposé dans [Dempster et al., 1977] pour être par la suite appliqué à l'apprentissage des paramètres d'un réseau bayésien dans [Lauritzen, 1995] et [Heckerman, 1995].

De la même manière que pour l'apprentissage à partir de données complètes, l'algorithme EM peut être appliqué selon une approche fréquentiste ou bayésienne.

3.3.1 Approche fréquentiste

Soit :
- $D_O = \{D_O^{(l)}\}_{l=1,...N}$, l'ensemble des instances de D pour lesquelles l'ensemble des valeurs prises par les variables du domaine sont observées.
- $\Theta^{(t)}$, l'ensemble des paramètres $\{\theta_{ijk}^{(t)}\}$ des paramètres du réseau bayésien $\mathcal{B}$, à l'itération t.

L'algorithme EM commence par estimer les valeurs des données manquantes (*espérance*) avant de les maximiser de la même manière que dans le cas complet (*maximisation*).

Algorithme 1 Algorithme EM pour l'apprentissage des paramètres d'un réseau bayésien

1: $t \leftarrow 0$

2: $\theta_{ijk}^{(0)} \leftarrow p > 0$ *(aléatoire)*

3: **Tant que** $|\theta_{ijk}^{(t+1)} - \theta_{ijk}^{(t)}| \geq \epsilon$ **Faire**

4: 1^e étape : Espérance

$$E(N_{ijk}) = \sum_{l=1}^{N} P(X_i = x_k | \Pi_i = \pi_{ij}, D_O^{(l)}, \theta_{ijk}^{(t)})$$

5: 2^e étape : Maximisation

$$\theta_{ijk}^{(t)} = \frac{E(N_{ijk})}{\sum_k E(N_{ijk})}$$

6: **Fin Tant que**

3.3.2 Approche bayésienne

Dans ce cas, nous employons des *a priori* de Dirichlet sur les paramètres. La différence avec le traitement fréquentiste réside dans la 2^e étape de l'algorithme 1 qui devient :

$$\theta_{ijk}^{(t)} = \frac{E(N_{ijk}) + \alpha_{ijk}}{\sum_k E(N_{ijk}) + \alpha_{ijk}} \tag{3.12}$$

Chapitre 4

Apprentissage de structures

Cette partie constitue une introduction générale à la problématique de l'apprentissage de la structure d'un réseau bayésien. Les algorithmes que nous décrivons ici ont pour objectif de trouver le réseau encodant le mieux la distribution de probabilité implicite à la base d'apprentissage qui leur est fournie en entrée. Le plan général de ce chapitre est celui employé dans [Naïm et al., 2004, François, 2006, Leray, 2006], Cependant, nous nous attardons volontairement sur certaines descriptions de méthodes ou de modèles tout en en négligeant d'autres, selon leur rapport avec nos travaux ou avec leur compréhension.

4.1 Généralités

La problématique de l'apprentissage de structure peut être comparée à l'exploration de données, c'est-à-dire à l'extraction d'une connaissance (dans notre cas, la topologie du réseau) à partir d'une base de données [Krause, 1999]. Si le deuxième chapitre présentait quelques usages des modèles bayésiens *déterminés*, nous pouvons néanmoins remarquer que dans certains cas, la détermination même du modèle peut constituer la problématique à résoudre. Ainsi, dans le cadre de la bio-informatique, les auteurs de [Yu et al., 2002] emploient l'apprentissage de la structure d'un réseau bayésien pour détecter les relations les plus évidentes entre différents régulateurs génétiques afin de pouvoir guider des expérimentations ultérieures. Dans ce type de problématique, la connaissance *a priori* du domaine que peut éventuellement avoir l'utilisateur ne permet que la détection d'incongruités flagrantes sur la structure obtenue automatiquement. La structure n'est plus alors seulement une partie de la solution au problème mais bien une solution à part entière.

Ce chapitre présente les méthodes les plus usitées dans le cadre de l'apprentissage de la structure, les hypothèses sous-jacentes ainsi que les avantages et inconvénients respectifs des différentes méthodes.

4.1.1 Cadre théorique

L'acquisition d'une structure représentative de la distribution de probabilité d'un domaine de variables présuppose certaines conditions (dont la condition locale de Markov, vue en 2.3.1). L'application de ces conditions et hypothèses permet de garantir l'existence d'une telle structure (*i.e.* que la distribution que nous cherchons à modéliser peut l'être par un *GOSC*) mais aussi de poser les bases sur lesquelles se fondent les différentes méthodes.

Hypothèse de fidélité

Nous avons d'ores et déjà supposé qu'un réseau bayésien vérifiait la condition locale de Markov : toute variable est indépendante de ses non-descendants connaissant ses parents (dans la structure $\mathcal{G}$ du réseau). La condition locale de Markov conjuguée à la d-séparation permet de déterminer un ensemble *minimal* d'indépendances conditionnelles dans le graphe $\mathcal{G}$. Cet ensemble est minimal car il peut exister des indépendances conditionnelles qui ne peuvent être déterminées par la seule lecture du graphe $\mathcal{G}$.

Prenons l'exemple de la figure 4.1 représentant un graphe $\mathcal{G}$ constitué de trois variables et deux arcs ainsi que les probabilités conditionnelles associées. On peut sans peine observer que, du fait du paramétrage particulier du modèle, X_1 est indépendant de X_2 connaissant X_3. Cette indépendance conditionnelle ne peut être relevée par la seule condition de Markov locale (et donc la d-séparation).

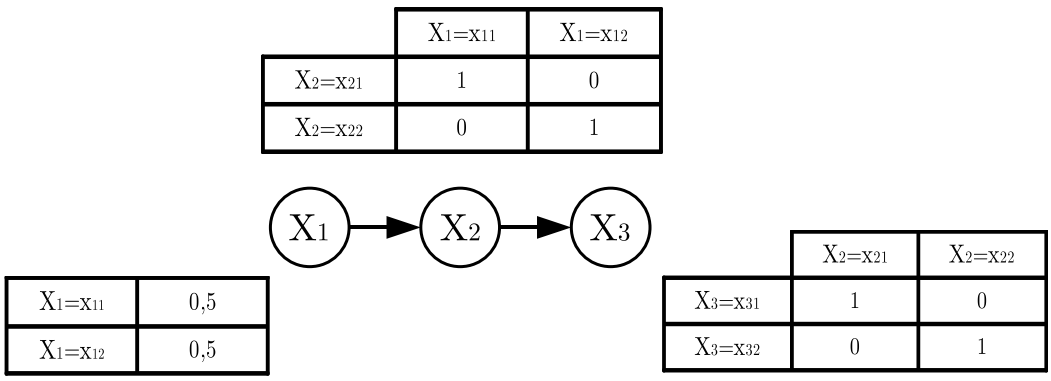


Figure 4.1 – Cas d'indépendance conditionnelle indétectable graphiquement.

Les indépendances conditionnelles impliquées par une distribution P, comme celle de la figure 4.1, ne peuvent être déterminées par l'application de la condition locale de Markov seule.

Or, il est dans notre intérêt est de pouvoir assurer la corrélation entre le graphe recherché et la distribution sous-jacente aux données à notre disposition. Pour cela, nous introduisons la notion de *fidélité*.

Définition 10 (Hypothèse de fidélité) *Une structure $\mathcal{G}$ et une distribution de probabilité P sont dites fidèles si et seulement si l'application de la condition de Markov permet de déduire l'ensemble des indépendances conditionnelles existant dans P et seulement celles-ci, i.e. :*

$$\forall X, Y \in \mathcal{U} \qquad X \perp\!\!\!\perp_P Y \Leftrightarrow X \perp_{\mathcal{G}} Y$$

En d'autres termes, l'hypothèse de fidélité appliquée à une distribution de probabilité P définie sur un domaine de variables $\mathcal{U}$ suppose l'existence d'une carte parfaite (ou P-map, cf. définition 8) du modèle d'indépendance associé à P.

Cette hypothèse invalide l'existence d'un modèle tel que celui de la figure 4.1.

Modularité paramétrique

Cette hypothèse était déjà émise dans le cadre de l'apprentissage de paramètres (section 3.1), nous pouvons la rappeler ici :

Définition 11 (Modularité paramétrique) *Soit deux GOSC $\mathcal{G}_1$ et $\mathcal{G}_2$, une variable $X_i \in \mathcal{U}$ et $\Pi_i(\mathcal{G})$, l'ensemble des sommets prédécesseurs du sommet X_i dans le GOSC $\mathcal{G}$:*
$\forall i \in 1 \ldots n$, si $\Pi_i(\mathcal{G}_1) = \Pi_i(\mathcal{G}_2)$, alors la densité de distribution des paramètres de X_i est la même pour $\mathcal{G}_1$ et $\mathcal{G}_2$.

Suffisance causale

Aussi appelée *hypothèse de complétude*. Nous supposons qu'il n'existe pas, dans le domaine modélisé, de variable non-observable qui soit parente de deux ou plus variables observées. L'ensemble des sommets suffit donc à représenter l'ensemble des relations pouvant être extraites des données observées. L'hypothèse de suffisance causale est cependant la plus à même de se voir invalider dans le cas de la modélisation de domaines non triviaux où l'informaticien ne peut garantir l'observation de l'ensemble des variables pertinentes au domaine.

Connaissance *a priori*

Certaines méthodes permettent de prendre en compte une connaissance *a priori* du modèle recherché, fournie alors par l'utilisateur. Certaines méthodes présentées ici permettent notamment l'apprentissage d'une structure à partir de la connaissance d'un ordre topologiquement compatible avec le modèle recherché.

Définition 12 (Ordre topologiquement compatible) *Un ordre topologiquement compatible avec un GOSC $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ est un ordre partiel $\prec$ sur les sommets de $\mathcal{G}$ tel que :*

$$\forall X \rightarrow Y \in \mathcal{E}, X \prec Y$$

Dans la suite, nous parlerons d'ordre topologique correct pour désigner un ordre topologiquement compatible avec le modèle sous-jacent aux données d'apprentissage.

4.1.2 Cadre pratique

L'apprentissage de structure d'un réseau bayésien doit éventuellement s'effectuer en tenant compte de la nature des données fournies pour l'apprentissage (ou simplement de la nature du domaine à modéliser) :

variables continues : les variables peuvent prendre leurs valeurs dans un espace continu (cf. section 2.5),

bases de données incomplètes : le cas des données incomplètes a déjà été évoqué dans le cadre de l'apprentissage des paramètres (section 3.2). Si, dans le cadre de nos travaux, nous traitons uniquement le cas des bases de données complètes, nous documenterons néanmoins l'apprentissage de la structure du modèle dans ce cas de figure,

insuffisance causale : il se peut que certaines variables du domaine observé soient conditionnellement dépendantes de variables non observées. Certains algorithmes, nous le verrons, permettent alors de détecter de telles variables.

Pour résumer, nos travaux ainsi que les méthodes présentées ici, sauf précision contraire, se réfèrent à l'apprentissage de structure dans le cas où :

- les variables modélisées prennent leurs valeurs dans un ensemble discret,
- les variables de la base d'apprentissage sont entièrement observées,
- il n'existe pas de variable latente (hypothèse de suffisance causale).

Dans la suite de ce chapitre, nous allons nous attacher à la présentation des différentes méthodes employables pour l'apprentissage de la structure d'un réseau bayésien.

Ces méthodes peuvent être réparties en deux principaux groupes :

Approche par découverte de relations d'indépendances : ces méthodes consistent en des procédures de tests sur les indépendances conditionnelles permettant, au final, de retrouver la structure recherchée.

Par exploration/évaluation : ces méthodes emploient un score afin d'évaluer la capacité du graphe à retranscrire les indépendances conditionnelles au sein du modèle.

C'est dans l'ordre de cette description que nous allons détailler le fonctionnement des principales méthodes.

4.2 Méthodes procédant par tests statistiques

Une manière de rechercher une structure adéquate pour un ensemble d'apprentissage est la recherche d'indépendances conditionnelles : la structure du réseau est déterminée pas à pas en établissant les indépendances conditionnelles existant au sein de l'ensemble des variables.
Si certains des algorithmes de ce type détectent des variables latentes (cachées), en revanche ils requièrent tous une base complète.

4.2.1 Algorithmes PC et IC

Le principe de l'algorithme PC (*Peter, Clark*, les prénoms des auteurs) est lui-même une évolution de l'algorithme SGS (*Spirtes, Glamour, Scheines*, tiré des noms de ses auteurs) [Spirtes et Scheines, 1991, Spirtes et al., 1993].

Le fonctionnement de l'algorithme PC est le suivant : Soit un graphe $\mathcal{G}(\mathcal{V}, E, \Theta)$, deux sommets X_i et X_j de $\mathcal{V}$ et un sous ensemble de sommets $S_{X_i,X_j} \in \mathcal{V}/\{X_i, X_j\}$. Les sommets X_i et X_j sont reliés par un arc dans $\mathcal{G}$ s'il n'existe pas S_{X_i,X_j} tel que $(X_i \perp\!\!\!\perp X_j | S_{X_i,X_j})$.

En pratique, la vérification de l'existence d'une telle indépendance revient à vérifier les différentes indépendances $(X_i \perp\!\!\!\perp_P X_j | S_{X_i,X_j})$ par ordre croissant (*ie.* selon une taille croissante de l'ensemble S_{X_i,X_j}. À partir d'un graphe non orienté entièrement connecté, la détection d'indépendances permet alors de supprimer les arêtes correspondantes jusqu'à l'obtention du squelette du *GOSC* recherché. Suivent alors deux phases distinctes visant dans un premier temps à détecter et établir les V-structures du graphe puis à orienter les arêtes restantes.

L'algorithme PC établit les hypothèses suivantes :
– hypothèse de fidélité (cf. section 4.1.1),
– la base d'apprentissage est complète et suffisamment grande,
– les résultats des tests d'indépendance conditionnelle sont fiables.
Il est courant d'employer le test du χ^2 ou bien celui du G^2 (cf. Annexes A).

À noter que l'algorithme PC, décrit dans l'algorithme 2, comme tous les algorithmes d'apprentissage de structure employant une base d'exemples, renvoie un graphe orienté appartenant à la classe d'équivalence de Markov du modèle recherché : les orientations des arcs, hormis celles des V-structures détectées, ne correspond pas forcément aux réels liens de causalité de ce modèle.

Les règles permettant ici d'orienter le graphe non-orienté obtenu à l'issue de la phase de détection des indépendances conditionnelles peuvent être remplacées par toute heuristique permettant l'obtention d'un *GOSC* à partir d'un tel graphe (comme, par exemple, l'algorithme de Dor et Tarsi [Dor et Tarsi, 1992]). Les deux opérations graphiques correspondant dans cette phase de l'algorithme PC à l'ajout d'un arc orienté reviennent simplement à orienter le graphe de manière à :
– ne pas créer de circuit ;
– de ne pas créer de V-structure ;
L'ordre dans lequel les variables sont alors considérées peut éventuellement déboucher sur des *GOSC* différents mais néanmoins représentant tous les mêmes ensembles d'indépendances conditionnelles (et donc *équivalents*, cf. section 4.4.4).

En revanche, l'ordre d'exécution des tests d'indépendance conditionnelle (*i.e.* l'ordre suivant lequel les différent sommets de l'ensemble conditionnant sont testés) a une influence sur le résultat final (exception faite du cas où la base d'apprentissage est de taille infinie) [Dash et Druzdzel, 1999].

En parallèle à l'algorithme PC, un autre algorithme, nommé IC (pour *Inductive Causation*) a été développé par l'équipe de Judea Pearl [Pearl et Verma, 1991]. Cet algorithme est similaire à l'algorithme PC mais part d'une structure vide en reliant les couples de variables dès qu'une dépendance conditionnelle est détectée (dans le sens où aucun sous-ensemble conditionnant S_{XY} tel que $(X \perp\!\!\!\perp Y)|S_{XY}$ n'est identifié) et obtient donc une D-map minimale là où l'algorithme PC cherche une I-map minimale (voir section 2.3.3).

L'inconvénient commun aux deux algorithmes est la multiplicité des tests d'indépendance conditionnelle à mener. Lors de la recherche de la structure d'un modèle complexe (nombreuses

Algorithme 2 Algorithme PC

Entrée: Un graphe connexe non orienté $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}, \mathcal{V} = \{X_1, X_2 \ldots, X_n\}$

1:
2: 1^e***étape : Déterminer les indépendances conditionnelles***
3: $k \leftarrow 0$;
4: $\mathcal{G} \leftarrow$ graphe complétement connecté;
5: $SepSet_{\mathcal{G}}(X_i, X_j) \leftarrow \emptyset, \forall (i, j) \in 1, \ldots, n$;
6: **Tant que** $\exists X_i, X_j \in V^2$ tq $|Adj(X_i)/X_j| \geq k$ **Faire**
7: ***Détermination des indépendances conditionnelles d'ordre k***
8: $\forall (X_i, X_j) \in \mathcal{V}^2$ tq $X_i - X_j$ et $|Adj(X_i)/X_j| \geq k$
9: Pour tout ensemble de sommets $S_{X_i,X_j} \subseteq Adj(X_i)/X_j$ tel que $|S_{X_i,X_j}| = k$
10: **Si** $X_i \perp\!\!\!\perp X_j | S_{X_i,X_j}$ **Alors**
11: $SepSet_{\mathcal{G}}(X_i, X_j) \leftarrow SepSet_{\mathcal{G}}(X_i, X_j) \cup S_{X_i,X_j}$ et supprimer l'arête $X_i - X_j$ dans $\mathcal{G}$
12: **Fin Si**
13: $k \leftarrow k + 1$;
14: **Fin Tant que**
15:
16: 2^e ***étape : Détection des V-structures***
17: **Pour** chaque triplet relié de $\mathcal{V}^3$ de la forme $X_i - Z - X_j$ **Faire**
18: **Si** $Z \notin SepSet_{\mathcal{G}}(X_i, X_j)$ **Alors**
19: Orienter : $X_i \rightarrow Z \leftarrow X_j$ dans $\mathcal{G}$
20: **Fin Si**
21: **Fin Pour**
22:
23: 3^e ***étape : Orientation des arêtes restantes***
24: **Tant que** $\nexists$ d'arête non orientée dans $\mathcal{G}$ **Faire**
25: $\forall (X_i, X_j) \in \mathcal{V}$
26: **Si** $X_i - X_j$ et $\exists$ un chemin orienté de X_i vers X_j **Alors**
27: orienter l'arête $X_i - X_j$ en $X_i \rightarrow X_j$
28: **Sinon**
29: **Si** $X_j \notin Adj(X_i), \forall Z$ tel que $X_i \rightarrow Z$ et $Z - X_j$ **Alors**
30: orienter l'arête $Z - X_j$ en $Z \rightarrow X_j$
31: **Fin Si**
32: **Fin Si**
33: **Fin Tant que**

variables), les deux algorithmes procèdent à des tests exhaustifs sur les différents ensembles conditionnels S_{X_i,X_j} possibles, pour chaque couple de sommets (X_i, X_j).

4.2.2 Algorithme BNPC

L'algorithme BNPC (pour *Bayes Net Power Constructor*) est décrit dans [Cheng et al., 2002] et utilise une analyse quantitative de l'information mutuelle entre les variables du domaine modélisé afin de construire la structure $\mathcal{G}$ recherchée. Les tests d'indépendance conditionnelle reviennent alors à déterminer un seuil pour l'information mutuelle (conditionnelle ou non) entre les couples de variables concernés.

BNPC se décompose en trois phases :

1- Élaboration : Un premier graphe $\mathcal{G}_1$ est créé par le même procédé que celui de l'algorithme MWST (voir section 4.4.1).

2- Enrichissement : Des arêtes sont ajoutées à $\mathcal{G}_1$ afin d'obtenir un graphe non-orienté $\mathcal{G}_2$ et ce, par application d'un nombre réduit de tests d'indépendance conditionnelle.

3- Affinement : Une nouvelle série de tests élimine les éventuelles arêtes superflues de $\mathcal{G}_2$ pour obtenir un graphe final, $\mathcal{G}$.

BNPC se décline sous deux variantes (BNPC-A et BNPC-B) selon que l'utilisateur fournisse ou non un ordre topologiquement compatible avec la structure recherchée. BNPC-A prend en entrée un tel ordre peut donc orienter les arêtes détectées à mesure de la construction.

Dans le cas de BNPC-B, cet ordre est inconnu et l'algorithme ne procède à l'orientation des différents arcs qu'au terme de son exécution.

La connaissance facultative d'un ordre topologique correct a pour conséquence une différence notable dans la manière dont les deux variantes construisent le graphe $\mathcal{G}$ recherché :

- BNPC-A peut détecter directement les différentes d-séparations du graphe à chaque étape de sa construction et donc définir précisément quels sont les différents ensembles de sommets conditionnants devant être pris en compte,
- BNPC-B, en l'absence d'un ordre topologique correct sur le graphe $\mathcal{G}$, se voit confronté au même problème que les algorithmes PC et IC à savoir la nécessité de tester un nombre exhaustif et donc exponentiel d'ensembles de sommets conditionnants afin de pouvoir déterminer si deux sommets X et Y doivent être ou non reliés par une arête.

Pour réduire sa complexité, BNPC-B diminue considérablement le nombre des ensembles de sommets conditionnants par l'intermédiaire d'une analyse quantitative des dépendances régnant au sein du graphe. Pour cela, les auteurs définissent l'hypothèse de la fidélité monotone.

Définition 13 (Fidélité monotone) *Soit $Chem_{\mathcal{G}}(X, Y)$, l'ensemble des chemins reliant les sommets X et Y dans un GOSC $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. Soit $Act_{\mathcal{G}}(X, Y|Z)$, l'ensemble des chemins de $\mathcal{G}$ activés par le sous ensemble de sommets Z et reliant les sommets X et Y, $\{X, Y\} \notin Z$. Soit $I(X, Y|Z)$ l'information mutuelle conditionnelle mesurée entre les sommets X et Y conditionnellement à Z. Alors $\mathcal{G}$ et la distribution de probabilité P sous-jacente aux données d'apprentissage sont monotonement fidèles si et seulement si :*

1. *$\mathcal{G}$ et P sont fidèles*

2. $\forall X, Y \in \mathcal{V}$, *si* $Act_{\mathcal{G}}(X, Y|Z') \subseteq Act_{\mathcal{G}}(X, Y|Z)$, *alors* $I(X, Y|Z') \leq I(X, Y|Z')$

En résumé, cette hypothèse établit que pour un *GOSC* $\mathcal{G}$ et une distribution de probabilités P fidèles, la quantité d'information transitant entre deux variables X et Y, conditionnellement à un sous ensemble de variables Z, est une fonction monotone du nombre de chemins reliant X à Y rendus actifs (cf. section 2.3.2) par Z. En prenant l'exemple de deux variables X et Y dont nous souhaitons tester l'indépendance, nous testons d'abord l'ensemble maximal C_M des variables pouvant former un ensemble d-séparant X et Y. Puis, en testant successivement et par cardinalité décroissante les sous-ensembles de C_M, nous pouvons obtenir un sous-ensemble de variables par lequel ne transite aucune information et d-séparant, donc, X et Y.

D'après les auteurs, il est alors possible de déterminer la structure $\mathcal{G}$ en $O(n^4)$.

Cependant, [Chickering et Meek, 2003] démontrent que l'hypothèse de fidélité monotone est raisonnablement incompatible avec l'hypothèse de fidélité (*i.e.* l'existence d'une P-map). Dans le cas où ces deux hypothèses coexistent (ce qui est le cas, dès lors que l'on suppose la fidélité monotone), alors survient une contradiction avec l'existence d'une chaîne au sein du graphe.

Définition 14 (Existence d'une chaîne dans un *GOSC*[Chickering et Meek, 2003]) *Un GOSC* $\mathcal{G}$ *possède une chaîne s'il comporte au moins l'une des deux configurations suivantes :*
- $X_1 \rightarrow X_2 \rightarrow X_3$
- $X_1 \leftarrow X_2 \rightarrow X_3$

De plus, [Chickering et Meek, 2003], toujours, démontrent que dans le cas précis où le *GOSC* recherché ne comporte aucune chaîne, alors la complexité des calculs requis par BNPC-B peut en fait être réduite à $O(n^2)$.

Ces dernières assertions mettent malheureusement en question la fiabilité de l'algorithme.

4.2.3 Commentaires

Les différents algorithmes procédant par recherche de causalité présentent des points communs. D'une part ces algorithmes présentent l'attrait de proposer une construction graduelle de la structure retournée. La prise en compte de propriétés graphiques locales aux différentes variables ainsi que l'emploi de méthodes statistiques connues rendent ce type d'approche intuitivement séduisante. Cependant, malgré ces traits intéressants, certains défauts demeurent :
- la fiabilité des tests d'indépendance, en particulier en présence d'un nombre de cas insuffisant ;
- le nombre important de tests d'indépendance à effectuer pour couvrir l'ensemble des variables ;
- dans le cas de l'algorithme BNPC, le manque de fiabilité de ses fondements théoriques.

Une alternative à l'apprentissage par le biais de tests statistiques est l'emploi d'une mesure d'évaluation de la qualité d'une structure vis-à-vis de la base d'apprentissage en combinaison avec une heuristique de parcours d'un espace de solutions candidates. La section suivante décrit quelques unes de ces méthodes.

4.3 Fonctions d'évaluation

Les méthodes procédant par exploration et évaluation des solutions potentielles utilisent un score permettant d'évaluer la concordance de la structure courante avec la distribution de probabilité ayant généré les données. De nombreuses fonctions d'évaluation ont été conçues et ce chapitre présente quelques-unes de celles-ci parmi les plus connues.

Certaines propriétés ont été déterminées comme sinon essentielles du moins d'importance pour les métriques employées. Ces propriétés sont les suivantes.

Décomposabilité Lors de l'emploi d'une heuristique parcourant l'espace des structures candidates, une transition depuis une structure $\mathcal{G}$ vers une structure $\mathcal{G}'$ se fait généralement à l'aide d'une opération du type ajout, soustraction ou inversion d'un arc. Dès lors qu'une modification ne modifie le calcul de la probabilité jointe du domaine que sur le terme $P(X_i|\Pi_i)$ où X_i est le nœud du graphe dont l'ensemble des parents a été modifié, il est intéressant de n'avoir à calculer l'impact de cette modification sur le score qu'en un terme dépendant de X_i et Π_i.

Définition 15 (Score décomposable) *Une fonction de score S est dite décomposable si, étant donné la structure $\mathcal{G}$ d'un réseau bayésien $\mathcal{B}$, $S(\mathcal{B})$ peut être exprimé sous la forme d'un produit (ou d'une somme, dans l'espace logarithmique) de scores locaux ne concernant qu'un sommet et ses parents.*

$$S(\mathcal{B}) = \sum_{i=1}^{n} s(X_i, \Pi_i) \text{ ou bien } S(\mathcal{B}) = \prod_{i=1}^{n} s(X_i, \Pi_i)$$

où n représente le nombre de sommets du graphe $\mathcal{G}$.

Cette propriété permet de réduire considérablement les coûts de calcul d'une heuristique de parcours de l'espace des *GOSC*.

Équivalence Deux *GOSC* différents peuvent encoder une même décomposition de la loi jointe sur le domaine modélisé. Les deux structures sont alors dites équivalentes *au sens de Markov* (cf. section 4.4.4 pour une définition complète de la notion d'équivalence). Il en résulte que plusieurs *GOSC* peuvent représenter le même ensemble d'indépendances conditionnelles et, par conséquent, présenter la même pertinence en terme de représentation de connaissance (dans le cas d'un graphe non-causal).

Définition 16 (Score équivalent) *Une fonction de score S est dite équivalente si, étant donné deux réseaux bayésiens $\mathcal{B}$ et $\mathcal{B}'$ équivalents au sens de Markov, S associe une même valeur aux structures de $\mathcal{B}$ et $\mathcal{B}'$.*

L'emploi d'un score équivalent permet donc de pouvoir distinguer des structures qui ne sont pas statistiquement équivalentes (et réciproquement, de regrouper les structures qui le sont). Dans le meilleur cas, nous devrions pouvoir ainsi déterminer une seule structure $\mathcal{G}$ statistiquement équivalente au graphe $\mathcal{G}^*$ recherché. Il nous reste à assurer l'existence de $\mathcal{G}^*$, ce qui est le rôle de la propriété de consistance.

Consistance Il est important de pouvoir garantir que la structure obtenant la meilleure évaluation soit celle du réseau à l'origine de la base de cas. Cette propriété est connue sous le nom de *consistance* du score employé.

Définition 17 (Score consistant) *Une fonction de score S est dite consistante si, lorsque la taille de la base d'apprentissage D tend vers l'infini, la structure $\mathcal{G}^*$ correspondant au modèle sous-jacent à D obtient le meilleur score avec une probabilité approchant 1.*

Par structure correspondant au modèle, nous entendons ici la structure qui est une carte d'indépendance minimale du modèle sous-jacent à D.

La plupart des scores que nous allons décrire par la suite possèdent ces propriétés. Les scores employés pour l'évaluation de structures peuvent eux-mêmes être répartis dans deux groupes : les scores dits bayésiens et les métriques fondées sur le principe de la longueur de description minimale.

Score bayésien

Si la dénomination de score bayésien réfère effectivement à une métrique spécifique que nous allons présenter ici, il s'agit aussi d'un terme plus général dénotant les différentes métriques développées à partir du même principe de base. Nous allons donc commencer par décrire ce principe avant de détailler ce qu'est le score bayésien à proprement parler ainsi que différentes variantes qui en ont été dérivées.

Lors de l'apprentissage d'un modèle, qu'il s'agisse de sa structure ou de ses paramètres, il existe une incertitude quant à l'identité de ces éléments. L'approche bayésienne consiste à représenter et quantifier cette incertitude sous une forme subjective. Cet encodage revient alors à déterminer une distribution *a priori* sur la structure et/ou les paramètres recherchés.

Nous cherchons ici la structure $\mathcal{G}$ ayant la probabilité la plus élevée conditionnellement aux données D. Autrement dit, nous cherchons à maximiser la probabilité $P(\mathcal{G}|D)$. Cette probabilité est la probabilité *a posteriori*. La maximisation de cette probabilité passe en premier lieu par sa décomposition ; le théorème de Bayes nous permet la décomposition suivante :

$$P(\mathcal{G}|D) = \frac{P(D,\mathcal{G})}{P(D)} = \frac{P(D|\mathcal{G}) \times P(\mathcal{G})}{P(D)} \tag{4.1}$$

Il est important de remarquer que l'élicitation du meilleur modèle se fait à partir de la seule base d'apprentissage D et que nous pouvons alors négliger la probabilité $P(D)$.

Le plus souvent, pour des raisons de commodité de calculs, l'optimisation de la probabilité *a posteriori* $P(D,\mathcal{G})$ passe par l'optimisation de son logarithme :

$$\log P(\mathcal{G}|D) = \log P(\mathcal{G}) + \log P(D|\mathcal{G}) - \log P(D) \approx \log P(\mathcal{G}) + \log P(D|\mathcal{G}) \tag{4.2}$$

Nous voyons que la probabilité $P(\mathcal{G}|D)$ se décompose en deux termes :

$P(\mathcal{G})$ **:** Probabilité *a priori* de la structure $\mathcal{G}$.

$P(D|\mathcal{G})$ **:** Vraisemblance marginale.

La probabilité *a priori* de $\mathcal{G}$, représente la mesure de la confiance (ou *a priori*) que nous avons en la structure $\mathcal{G}$.

La valeur de la vraisemblance marginale $P(D|\mathcal{G})$ a été calculée quant à elle, dans le cas de variables discrètes, par [Cooper et Herskovits, 1992] qui ont proposé le résultat suivant :

Théorème 2 *Soit :*
- *U, un ensemble de variables aléatoires discrètes, $U = \{X_1, X_2, \dots X_n\}$;*
- *$X_i, i \in 1 \dots n$ variable de U de cardinalité r_i ;*
- *D, base d'apprentissage formée de N instances de $\mathcal{U}$ indépendantes et identiquements distribuées ;*
- *$\mathcal{G}$ une structure d'un réseau bayésien contenant exactement les n variables de $\mathcal{U}$;*
- *π_{ij}, j^e instanciation de Π_i, $j \in 1 \dots q_i$;*
- *N_{ijk} le nombre d'occurrences simultanées de $X_i = x_i^k$ et π_{ij} ;*
- *soit $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.*

Le score bayésien, aussi appelé score BD, s'écrit alors :

$$Score_{BD}(\mathcal{G}, D) = P(\mathcal{G}, D) = P(\mathcal{G}) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \tag{4.3}$$

Le score bayésien de l'équation 4.3 s'adapte au cas de l'utilisation d'*a priori* de Dirichlet (cf. section 3.3.2) en s'écrivant :

$$Score_{BD}(\mathcal{G}, D) = P(\mathcal{G}, D) = P(\mathcal{G}) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(\alpha_{ij} + r_i - 1)!}{(N_{ij} + \alpha_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} \frac{(\alpha_{ijk} + N_{ijk})!}{\alpha_{ijk}!} \tag{4.4}$$

La distribution *a priori* sur l'espace des structures peut être définie ou calculable grâce à un expert ou bien, cas le plus courant car simplifiant le calcul, être définie comme étant uniforme. Dans ce dernier cas, la maximisation de la probabilité relative *a posteriori* $\log P(D, \mathcal{G})$ se ramène alors à la maximisation de la vraisemblance marginale que l'on emploie comme score.

Le score BDe

Le score bayésien, présenté précédemment, présente un inconvénient majeur : il n'est pas équivalent (cf. définition 16). Deux structures présentant les mêmes indépendances conditionnelles obtiennent donc deux évaluations différentes. Une heuristique parcourant l'espace des structures a tout intérêt à employer un score présentant la propriété d'équivalence.

À cette fin, les auteurs de [Heckerman et al., 1995a] ont développé une variante du score BD. Cette variante repose sur l'hypothèse d'équivalence de la vraisemblance :

Hypothèse 1 (Équivalence de vraisemblance) *Soit $\mathcal{G}_1$ et $\mathcal{G}_2$, deux structures représentant les mêmes indépendances conditionnelles, de probabilités a priori non négatives, alors $P(\Theta|\mathcal{G}_1) = P(\Theta|\mathcal{G}_2)$.*

Soit $\mathcal{G}_T$, le graphe entièrement connecté sur $\mathcal{V}$ et N_{est}, un nombre arbitraire de pseudo-exemples supplémentaires – *i.e.* un décompte fictif d'exemples supplémentaires de la base pour lesquels $X_i = x^k, \Pi_i = \pi_{ij}$ –, la contrainte suivante, imposée sur les exposants de Dirichlet α_{ijk} des distributions des paramètres du modèle évalué, permet de rendre le score BD équivalent :

$$\alpha_{ijk} = N_{est} \times P(X_i = x_k, \Pi_i = \pi_{ij} | \mathcal{G}_T) \tag{4.5}$$

L'avantage de fixer un nombre minimal N_{est} d'occurrences pour les différentes configurations possibles est d'empêcher qu'une configuration particulière d'une variable X_i et de son ensemble de variables parents Π_i ne soit considérée comme impossible par le modèle (et se voit donc attaché une probabilité *a posteriori* nulle).

La variante, nommée score BDe (pour *Bayesian Dirichlet Equivalent*), est alors égale à :

$$S_{BDe}(\mathcal{G}|D) = P(\mathcal{G}, D) = P(\mathcal{G}) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(N_{ij} + \alpha_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})} \tag{4.6}$$

La fonction gamma Γ étant employée ici du fait que les α_{ijk} ne sont pas nécessairement entiers.

Le score BDeu

Le score BDeu est en fait un cas particulier du score BDe présenté précédemment. Ici, les distributions de probabilités *a priori* définies sur les paramètres du modèle évalué sont uniformes (le 'u' de BDeu signifiant *uniform*), *i.e.*

$$P(X_i = x_k, \Pi_i = \pi_{ij} | \mathcal{G}_T) = \frac{1}{r_i . q_i}, \quad \forall i \in 1 \dots N, k \in 1 \dots r_i, j \in 1 \dots q_i$$

$\mathcal{G}_T$ représentant ici encore le graphe entièrement connecté sur $\mathcal{V}$.

[Buntine, 1991] et [Heckerman et al., 1995a] définissent alors les exposants de Dirichlet suivants (cf. equation 4.5) :

$$\alpha_{ijk} = \frac{N_{est}}{r_i q_i}, \quad \forall i \in 1 \dots N, k \in 1 \dots r_i, j \in 1 \dots q_i \tag{4.7}$$

où N_{est} désigne, ici encore, un nombre arbitrairement fixé.

L'équation 4.6 désigne alors le score BDeu, qui demeure un score équivalent.

Si $N_{est} = 1$, le score BDeu est alors égal à :

$$S_{BDeu}(\mathcal{G}|D) = P(\mathcal{G}) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\frac{1}{q_i})}{\Gamma(\frac{1}{q_i} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(\frac{1}{q_i r_i} + N_{ijk})}{\Gamma(\frac{1}{q_i r_i})} \tag{4.8}$$

Le score BDeu est très souvent employé pour l'évaluation de structures. Une de ses caractéristiques est d'être apte à détecter les arcs correspondant à des dépendances conditionnelles

de poids faibles. Ce comportement est d'ailleurs d'autant plus prononcé que l'on emploie une valeur élevée pour le paramètre N_{est} de l'équation 4.7 [Kayaalp et Cooper, 2002].

La vraisemblance du modèle évalué par rapport à la base d'apprentissage n'est pas la seule méthode d'évaluation d'une structure. Une autre façon de faire est d'employer des critères visant non seulement à vérifier cette adéquation mais aussi, dans le cas des fonctions de scores que nous allons présenter, de favoriser les modèles les plus simples.

Le critère AIC

Le critère AIC (*Akaike Information Criterion*) [Akaike, 1970] cherche à éviter les problèmes liés à l'apprentissage sur la seule vraisemblance. Dans les évaluations précédentes, la vraisemblance par rapport à la base de données est employée à la fois pour l'apprentissage des paramètres du modèle et pour l'évaluation de la structure, ce qui risque d'introduire un biais.

En pénalisant la complexité des structures évaluées, le critère AIC vise à éliciter le modèle le plus simple et le plus expressif de la connaissance extraite de la base D.

$$ScoreAIC(\mathcal{B}, D) = -2\log L(D|\mathcal{B}, \Theta^{MV}) + 2Dim(\mathcal{B}) \tag{4.9}$$

Où Θ^{MV} est l'ensemble des paramètres obtenus par maximum de vraisemblance pour le réseau bayésien $\mathcal{B}$ et $Dim(\mathcal{B})$ est la dimension de $\mathcal{B}$.
Soit r_i le nombre de valeurs pouvant être prises par la variable X_i, le nombre de paramètres nécessaires pour représenter $P(X_i|\Pi_i = \pi_{ij})$ est égal à $r_i - 1$ et la représentation de $P(X_i|\Pi_i)$ nécessite $Dim(X_i, \mathcal{B})$ paramètres avec :

$$Dim(X_i, \mathcal{B}) = (r_i - 1) \times q_i, \text{ où } q_i = \prod_{X_j \in \Pi_i} r_j$$

La dimension du modèle $\mathcal{B}$ devient alors :

$$Dim(\mathcal{B}) = \sum_{i=1}^{n} Dim(X_i, \mathcal{B})$$

AIC, bien que décomposable et équivalent, présente cependant un inconvénient : celui de ne pas être consistant avec la dimension. Il faut comprendre par là que le critère AIC ne permet pas la sélection du véritable modèle (à supposer qu'il existe) quand la taille de la base d'apprentissage s'accroît. En effet, lorsque la taille de la base croît, le terme de pénalisation tend à devenir négligeable vis-à-vis de la vraisemblance logarithmique. Dès lors, le critère AIC va avoir tendance à éliciter le modèle le plus complexe et aboutir ainsi à un surapprentissage.

Pour circonvenir à ceci, diverses variantes du critère AIC ont été développées parmi lesquelles le critère CAIC (pour *Consistent AIC*) [Bozdogan, 1987].

$$Score_{CAIC}(\mathcal{B}, D) = -2\log L(D|\mathcal{B}, \Theta^{MV}) + Dim(\mathcal{B}) \times [\log(N) + 1] \tag{4.10}$$

À l'opposé, dans le cas où la taille de la base de données est très limitée, il est généralement préférable d'employer une autre variante du critère AIC : le critère AICC (*Akaike Information Corrected Criterion*)[Hurvich et Tsai, 1989]. Cette variante inflige une pénalité d'autant moins élevée aux structures complexes que la base est limitée, relâchant la contrainte de parcimonie.

$$Score_{AICC}(\mathcal{B}, D) = AIC + \frac{2Dim(\mathcal{B}) \times (Dim(\mathcal{B}) + 1)}{N - Dim(\mathcal{B}) - 1} \tag{4.11}$$

Pour l'équation 4.11, nous remarquons que, lorsque la taille de la base de données devient importante, le dernier terme de l'équation tend vers 0 et on approche la formule du critère AIC.

Le score MDL

Le critère MDL [Rissanen, 1978, Suzuki, 1996] incorpore un terme pénalisant les structures trop complexes et ne tient pas seulement compte de la complexité même du modèle mais aussi de la complexité du codage des données suivant ce modèle.

Il existe différentes variantes du score MDL, dont celle de [Lam et Bacchus, 1994]

$$Score_{MDL}(\mathcal{B}, D) = LL(D|\Theta^{MV}, \mathcal{B}) - |\mathcal{E}_{\mathcal{B}}| \times logN - c.Dim(\mathcal{B}) \tag{4.12}$$

$|\mathcal{E}_{\mathcal{B}}|$ représentant le nombre d'arcs présents dans le modèle $\mathcal{B}$ et c, le nombre de bits nécessaires pour encoder un des paramètres.

Le score BIC

Le critère BIC (*Bayesian Information Criterion*), proposé dans [Schwartz, 1978], est certainement le critère le plus employé actuellement dans le cadre de la sélection de modèles.
Semblable au critère AIC, il comporte lui aussi une pénalité envers la complexité structurelle.

$$ScoreBIC(\mathcal{B}, D) = \log L(D|\mathcal{B}, \Theta^{MAP}) - \frac{1}{2} Dim(\mathcal{B}) \times \log N \tag{4.13}$$

Le terme $-\frac{1}{2} Dim(\mathcal{B}) \times \log N$ représentant la pénalité envers les structures trop complexes. Si l'on évalue un ensemble $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$ de modèles, le meilleur modèle M_i est alors égal à :

$$M = argmax_{M_i}(scoreBic) \tag{4.14}$$

Le score BIC présente les trois propriétés présentées en début de section, à savoir l'équivalence, la décomposabilité et la consistance ; ceci, combiné à sa tendance à éliciter les modèles les plus simples [Bouckaert, 1994] en fait une des métriques d'évaluation les plus employées avec le score BDeu.

Commentaires

Nous avons vu, dans cette section, plusieurs fonctions de score. Il s'avère, dans la pratique, que la plupart des méthodes actuelles se fondent sur l'emploi du score BDeu ou bien du critère BIC. Certaines méthodes emploient des versions modifiées des critères pondérés comme AIC ou MDL.
Les algorithmes que nous allons présenter emploient indifféremment n'importe quelle fonction d'évaluation sauf mention contraire (certaines méthodes requièrent explicitement une évaluation ayant la propriété d'équivalence, par exemple).

4.4 Algorithmes employant un score

Cette approche consiste à parcourir l'ensemble des structures envisageables (au sens le plus large : l'espace des *GOSC* décrits sur l'ensemble des variables modélisées), à évaluer celles-ci puis à renvoyer la structure ayant obtenu la meilleure évaluation. Une telle stratégie se heurte cependant au problème de la taille de l'espace à parcourir. Soit $r(n)$, le nombre de graphes orientés sans circuit réalisables sur un ensemble de n sommets :

$$r(n) = \sum_{i=1}^{n} (-1)^{i+1} C_i^n 2^{i(n-i)} r(n-i) = n^{2^{O(n)}} \tag{4.15}$$

L'équation 4.15 a été démontrée dans [Robinson, 1976].
Nous voyons que si pour $n = 4$, la valeur $r(4) = 543$ demeure raisonnable, dès que $n = 7$ le nombre de graphes à parcourir devient impressionnant : $r(7) \approx 1,4 \times 10^9$, $r(12) \approx 5,2 \times 10^{26}$.

Dès lors, il est nécessaire de mettre au point des heuristiques permettant de parcourir efficacement cet espace.

Ces heuristiques parcourent généralement l'espace des *GOSC* suivant une heuristique afin de restreindre l'espace de recherche ; il existe cependant, comme nous le verrons, des alternatives à l'espace des *GOSC*, tel que l'espace des arbres ou encore, celui de graphes servant à représenter les classes d'équivalence au sens de Markov des structures.

4.4.1 Recherche de l'arbre de recouvrement de poids maximal

La recherche de l'arbre de recouvrement maximal dans un graphe pondéré peut être adaptée à l'apprentissage de structure. Dans [Chow et Liu, 1968], les auteurs associent à chaque couple (X_i, X_j) de variables un poids - dans ce cas précis l'information mutuelle entre X_i et X_j. Une autre possibilité de pondération consiste à remplacer l'information mutuelle par un score local en X_i et X_j [Heckerman et al., 1995a].

L'application à la recherche de structure se fait alors de la manière suivante :

1. recherche de l'arbre (non-orienté) de recouvrement maximal ;
2. orientation de l'arbre obtenu.

La première étape est achevée par une heuristique standard telle que l'algorithme de Kruskal [Cormen et al., 1994]. La deuxième phase est, elle, accomplie en choisissant tout d'abord un nœud faisant office de racine puis en orientant les arêtes restantes à partir de celui-ci.

Les avantages de cet algorithme sont nombreux. Aussi simple que rapide, il force de plus l'appartenance de chaque variable du domaine à la structure retournée ; augmentant les chances de détecter des relations faibles qui sont ignorées dans le cas contraire. Les inconvénients sont que le choix du nœud racine est arbitraire en l'absence d'*a priori* de même que le fait de relier obligatoirement l'ensemble des variables entre elles peut aussi générer des relations inopportunes en reliant des variables normalement conditionnellement indépendantes.

De la même manière, dans certaines problématiques, la recherche de la structure bayésienne optimale a pour but de permettre de cartographier les (in)dépendances conditionnelles du domaine ; MWST empêche toute variable dont la représentation serait superflue d'être détectée comme telle. Il serait alors nécessaire de sortir de l'espace des arbres pour passer dans celui des forêts.

4.4.2 Algorithme K2

A l'origine, l'algorithme K2 [Cooper et Herskovits, 1992] évalue les différentes solutions à l'aide du score bayésien, BD (cf. section 4.3).

L'espace des *GOSC* étant de taille exponentielle par rapport au nombre de variables modélisées (voir section 4.4), l'algorithme K2 va réduire l'espace de recherche en prenant en entrée un ordre topologique correct sur les variables du domaine.
La réduction qui découle de cette connaissance permet de limiter la recherche à un espace de $2^{C_n^2}$ structures.
De même, l'espace à explorer est aussi limité en faisant l'hypothèse qu'un nœud ne peut avoir plus d'un certain nombre de parents.
Un algorithme de recherche glouton sur les ensembles de parents potentiels de chaque nœud est alors employé.

L'heuristique de parcours de K2 peut être employée, comme cela l'a été précédemment évoqué, avec une autre mesure que le score BD, telle que le critère MDL [Bouckaert, 1993].

4.4.3 Algorithme Greedy Search

L'algorithme Greedy Search [Chickering et al., 1995] (ou algorithme glouton) parcourt l'espace des GOSC à l'aide de successions d'opérations élémentaires. À chaque itération, l'algorithme explore le voisinage d'une structure candidate à une opération d'édition près, où une opération consiste en un ajout, une soustraction ou une inversion d'arc. Si une des structures voisines obtient un meilleur score que la structure candidate, elle remplace cette dernière et l'algorithme réitère l'exploration. La terminaison survenant dès lors qu'aucune structure voisine n'obtient un meilleur score.

Algorithme 3 K2

Entrée: Un ensemble de n noeuds, un ordonnancement sur ces noeuds, une borne supérieure S sur le nombre de parents d'un noeud, une base de données D composée de N cas

Pour $i = 1 \dots n$ **Faire**
  $\pi_i \leftarrow \emptyset$;
  $P_{prec} \leftarrow f(X_i, \Pi_i)$;
  continuer $\leftarrow$ *VRAI*;
  Tant que *continuer* et $|\pi_i| < S$ **Faire**
    soit z le noeud de $Pred(X_i)$ maximisant $Score_{BD}(X_i, \Pi_i \cup \{z\})$;
    $P_{nouveau} \leftarrow f(i, \pi_i \cup \{z\})$;
    Si $P_{nouveau} > P_{prec}$ **Alors**
      $P_{prec} \leftarrow P_{nouveau}$;
      $\Pi_i \leftarrow \Pi_i \cup \{z\}$;
    Sinon
      continue $\leftarrow$ *FAUX*;
    Fin Si
  Fin Tant que
Fin Pour

4.4.4 Recherche gloutonne sur l'espace des graphes essentiels

Une alternative à la recherche de structures sur l'espace des *GOSC* consiste à identifier non plus une structure mais un graphe représentant la classe d'équivalence au sens de Markov (cf. section 4.4.4) de celle-ci.

Nous commençons par décrire un tel graphe ainsi que ses propriétés avant de décrire une méthode gloutonne sur l'espace de ces graphes représentants, l'algorithme *GES*.

Équivalents de Markov

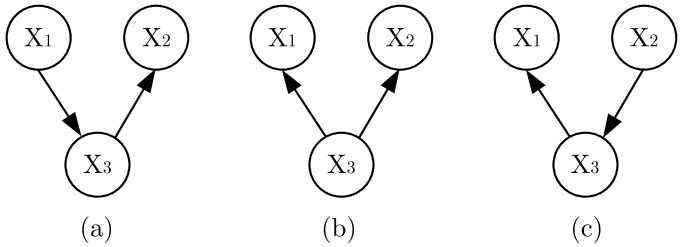


Figure 4.2 – Exemples simples illustrant la notion d'équivalence. Les graphes (a), (b) et (c) encodent la même décomposition de loi jointe.

Plusieurs graphes orientés sans circuits peuvent encoder les mêmes ensembles d'indépendances conditionnelles.

Définition 18 *Deux réseaux bayésiens $\mathcal{B}_1$ et $\mathcal{B}_2$ sont dits équivalents au sens de Markov si leurs structures encodent la même décomposition de la loi jointe.*

Le critère de d-séparation permet la lecture des indépendances conditionnelles encodées par le graphe. Par la suite, nous emploierons le terme général d'équivalence pour désigner l'équivalence au sens de Markov.

La figure 4.2 illustre l'exemple de trois graphes différents représentant néanmoins le même ensemble d'indépendances conditionnelles. Cette équivalence se démontre simplement :

Démonstration 2 *Le graphe (a) encode :*

$$P(X_1, X_2, X_3) = P(X_1) \times P(X_3|X_1) \times P(X_2|X_3) = P(X_3, X_1) \times P(X_2|X_3) \quad (4.16)$$
$$= P(X_1|X_3) \times P(X_3) \times P(X_2|X_3) \quad (4.17)$$

Donc le graphe (a) et le graphe (b) encodent la même loi de probabilité jointe. La même démarche peut s'appliquer au graphe (c).

Considérons maintenant la figure 4.3.

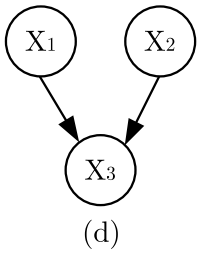


Figure 4.3 – Exemple de V-structure.

La décomposition de sa loi jointe est la suivante :

$$P(X_1, X_2, X_3) = P(X_3|X_1, X_2) \times P(X_1) \times P(X_2).$$

Contrairement aux autres graphes, nous avons ici un terme $P(X_3|X_1, X_2)$ ne pouvant pas se simplifier. Le graphe de la figure 4.3 encode simultanément les (in)dépendances conditionnelles $(X_1 \perp\!\!\!\perp X_2)$ et $(X_1 \not\perp\!\!\!\perp X_2|X_3)$. Ce graphe possède une forme caractéristique que nous avons déjà entraperçue dans la définition de la d-séparation : il s'agit d'une V-structure. Nous avons à notre disposition le théorème suivant :

Théorème 3 ([Verma et Pearl, 1990]) *Deux réseaux bayésiens sont dits équivalents au sens de Markov si et seulement si leurs structures ont le même squelette et les mêmes V-structures.*

où le squelette d'un graphe orienté $\mathcal{G}$ désigne le graphe non orienté obtenu en ignorant les orientations de ses arcs.

Les structures représentant la même décomposition de la loi jointe sur un domaine donné appartiennent à une même classe d'équivalence. Il est alors possible de représenter cette classe d'équivalence à l'aide d'un graphe partiellement orienté sans circuit (ou *GPOSC*) appelé *graphe essentiel* (ou *GE*) [Andersson et al., 1995].

Définition 19 *Pour un GOSC $\mathcal{G}$, le graphe partiellement orienté sans circuit obtenu en ignorant l'orientation des arcs réversibles de $\mathcal{G}$ est appelé graphe essentiel de $\mathcal{G}$. Ce graphe est le représentant de la classe d'équivalence de $\mathcal{G}$.*

Définition 20 *Un arc est dit réversible s'il n'appartient à aucune V-structure et si son inversion ne crée ou ne détruit aucune V-structure.*

Des algorithmes existent permettant d'obtenir le graphe essentiel, ou *GE*, d'une structure donnée [Chickering, 1996] ou bien, à l'inverse, d'obtenir une instanciation sous forme d'un *GOSC* d'un graphe partiellement orienté (et donc d'une instanciation d'un *GE*) [Chickering, 1995, Dor et Tarsi, 1992].

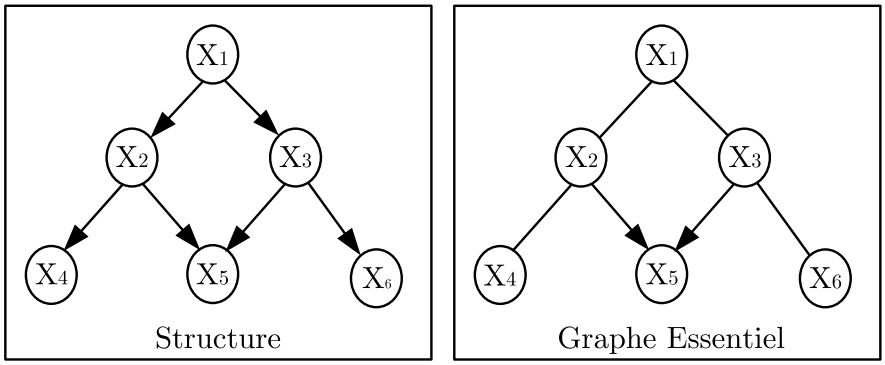


Figure 4.4 – Exemple de graphe orienté sans circuit et de son graphe essentiel.

La figure 4.4 montre côte à côte le *GOSC* vu précédemment dans la figure 2.7 et le *GE* correspondant. Nous voyons que seule la V-structure centrale demeure orientée.

D'après [Perlman et Gillispie, 2001], le ratio entre le nombre de *GE* pour un nombre donné de variables modélisées et le nombre de *GOSC* correspondant est asymptotiquement égal à 3,7. Ce résultat a été observé pour un nombre de variables allant jusqu'à 10.

Enfin, il est intéressant de préciser ici que des graphes équivalents au sens de Markov proposent la même décomposition de la loi de probabilité jointe sur le domaine et, par conséquent, l'orientation des arcs n'appartenant pas à des V-structures, bien que sans rapport avec les réels liens de causalité au sein du domaine, n'a pas d'influence sur les processus d'inférence vus en section 2.6.

Algorithme GES

Consécutivement à la définition de l'espace des représentants des classes d'équivalence de Markov, plusieurs travaux se sont employés à travailler dans cet espace. Les arguments allant en faveur d'une telle approche étant :

Taille de l'espace : plusieurs *GOSC* représentant les mêmes indépendances conditionnelles peuvent être représentés par un seul et unique *GE* ;

Non-redondance : un *GE* représentant une et une seule classe d'équivalence, toute modification à un *GE* entraîne une réelle modification de l'espace exploré.

Les principaux travaux en matière d'exploitation de l'espace des *GE* sont dus à Chickering [Chickering, 2002b]. L'algorithme développé par l'auteur est un glouton sur l'espace des *GE* procédant en deux phases distinctes :

- une phase d'ajouts : en partant d'un graphe d'origine quelconque, cette phase ajoute successivement les arcs manquants et correspondant aux dépendances conditionnelles détectées dans la base d'apprentissage ;
- une phase de retraits d'arcs : suite à la première phase, cette phase retire successivement les arcs superflus.

L'algorithme GES est optimal sous condition que les hypothèses suivantes soient vérifiées :
- les exemples contenus dans la base d'apprentissage D employée sont indépendants et identiquement distribués ;
- D est suffisament grande pour que la distribution de probabilités qu'elle définit soit égale à la distribution de probabilité p définie sur le domaine de variables $\mathcal{U}$ par le modèle que nous recherchons ;
- le score utilisé est consistant (cf. définition 17) ;
- enfin, que la distribution représentée par D soit fidèle à un *GOSC* (cf. définition 10).

Le principal fondement théorique de l'optimalité de cet algorithme est la conjecture de Meek [Meek, 1997], émise par Christopher Meek. Cette conjecture a été partiellement démontrée dans [Kocka et al., 2001] avant d'être démontrée dans sa totalité par Chickering [Chickering, 2002a].

Théorème 4 (Conjecture de Meek) *Soit $\mathcal{G}_1$ et $\mathcal{G}_2$, deux GOSC où $\mathcal{G}_2$ est une carte d'indépendances de $\mathcal{G}_1$. Alors il est possible à l'aide d'une séquence $\mathcal{S}$ finie d'opérations d'ajout et d'inversion des arcs de $\mathcal{G}_1$ telle que :*

1. *à la suite de toute opération élémentaire, $\mathcal{G}_1$ est un GOSC et $\mathcal{G}_2$ est toujours une carte d'indépendances de $\mathcal{G}_1$,*
2. *après avoir appliqué l'ensemble des opérations de $\mathcal{S}$, $\mathcal{G}_1 = \mathcal{G}_2$.*

Cette conjecture signifie concrètement que, dès lors que l'on suppose que la distribution de probabilité recherchée peut être représentée par un *GOSC*, alors on peut retrouver une carte d'indépendances (un graphe où sont représentées toutes les dépendances de la distribution modélisée) à partir d'une carte de dépendances (un graphe où sont représentées toutes les indépendances de la distribution modélisée).

Les auteurs définissent deux voisinages successivement explorés par GES : l'inclusion supérieure et l'inclusion inférieure [Chickering, 2002b].

Définition 21 *Soit $\mathcal{G}_E$, un GE, la limite d'inclusion supérieure $V^+(\mathcal{G}_E)$ est l'ensemble des GE voisins de $\mathcal{G}_E$ tels que :*

$$\mathcal{G}_E^+ \in V^+(\mathcal{G}_E) \; ssi \; \exists \mathcal{G} \equiv \mathcal{G}_E / \mathcal{G}^+ = \{\mathcal{G} + 1 \; arc\} \; et \; \mathcal{G}^+ \equiv \mathcal{G}_E^+$$

Définition 22 *Soit $\mathcal{G}_E$, un GE, la limite d'inclusion inférieure $V^-(\mathcal{G}_E)$ est l'ensemble des GE voisins de $\mathcal{G}_E$ tels que :*

$$\mathcal{G}_E^- \in V^-(\mathcal{G}_E) \; ssi \; \exists \mathcal{G} \equiv \mathcal{G}_E / \mathcal{G}^- = \{\mathcal{G} - 1 \; arc\} \; et \; \mathcal{G}^- \equiv \mathcal{G}_E^-$$

Pour explorer les différents voisinages d'inclusion ($V^+(\mathcal{G}_E)$ et $V^-(\mathcal{G}_E)$), GES opère en deux phases [Chickering, 2002b] :

Sur $V^+(\mathcal{G}_E)$: sont considérés, à chaque itération de cette phase, tous les graphes du voisinage du graphe actuel obtenus par ajout ou inversion d'un arc. Pour cette dernière opération, ne sont considérées que les inversions impliquant un changement de classe d'équivalence. À chaque itération, le nouveau graphe sélectionné est celui ayant entraîné la plus grande amélioration du score.

Sur $V^-(\mathcal{G}_E)$: cette phase est similaire à la précédente, la différence étant qu'elle opère par soustractions et inversions successives jusqu'à ce que le score ne puisse plus être amélioré.

Le fait que GES explore l'espace des *GE* à l'aide d'une fonction de score implique bien entendu que ce score soit équivalent. L'initialisation de GES peut se faire à partir d'un graphe quelconque, même s'il est généralement convenu de démarrer avec le graphe vide $\mathcal{G}$. Il est de même possible, bien que l'algorithme que nous venons de décrire corresponde à la version originale de GES telle que décrite dans [Meek, 1997], d'employer conjointement à chaque itération le voisinage supérieur et le voisinage inférieur. Cette variante, présentée dans [Chickering, 2002a] est elle aussi, asymptotiquement optimale.

À noter que l'exploitation de l'espace des *GE* présente cependant certains inconvénients :

- La taille de l'espace n'est pas aussi réduite que l'on pourrait l'espérer : il a été démontré, pour un nombre de variables n<10, que le quotient de la taille de l'espace des *GOSC* avec celle de l'espace des GE était asymptotiquement égal à 3,7 [Perlman et Gillispie, 2001, Gillispie et Perlman, 2002] ;
- L'affranchissement de l'espace des *GOSC* n'est pas total, il est nécessaire pour pouvoir évaluer la qualité d'une classe d'équivalence d'en instancier le représentant sous la forme d'un *GOSC*.

[Nielsen et al., 2003] introduisent KES qui est une variante de GES pour laquelle la gloutonnerie est relâchée : plutôt que de sélectionner à chaque étape le meilleur graphe du voisinage, KES sélectionne aléatoirement un graphe améliorant la solution en cours.

4.4.5 Commentaires

De la même manière que nous avons commenté les méthodes procédant par détection des indépendances conditionnelles, nous pouvons ici faire un constat des forces et faiblesses des algorithmes vus dans cette section.

- les algorithmes de parcours permettent d'obtenir des solutions caractérisées par la fonction de score employée (modèle plus simple ou au contraire tenant compte des dépendances les plus faibles,...). Ceci, pour le cas d'apprentissages à partir de données restreintes, les scores étant pour la plupart asymptotiquement égaux pour une quantité de données d'apprentissage suffisante ;
- le développement de nouvelles heuristiques se fonde principalement sur le parcours de l'espace de recherche et la discrimination de bonnes solutions ce qui en fait une approche séduisante ;
- la découverte de la structure optimale à partir d'une fonction de score est NP-difficile [Chickering et al., 1995, Chickering et al., 1994, Chickering et al., 2003] ;

– certaines heuristiques (GES, GS) sont coûteuses en temps de calcul.

Malgré les difficultés inhérentes à l'utilisation d'une fonction d'évaluation pour trouver une bonne structure, l'aspect même de la problématique suscite encore aujourd'hui de nombreux travaux ayant pour objectif d'optimiser le parcours de l'espace de recherche.

4.4.6 Méthodes hybrides

Certaines méthodes tentent de combiner les avantages des deux méthodologies que nous venons de voir. Un certain nombre de ces méthodes s'inscrivent dans l'application d'un algorithme évolutionnaire – nous le verrons plus loin –. Le principe de ces méthodes consiste généralement à limiter l'espace de la recherche effectuée par un algorithme employant un score. Cette limitation est définie à partir de tests d'indépendance conditionnelle d'ordre peu élevé – 1 ou 0 – et permet d'éviter la perte de temps causée par l'évaluation de solutions inintéressantes [van Dijk et al., 2003b].

L'inconvénient d'une telle approche est que, non contente de combiner les qualités des deux approches, elle en combine aussi les défauts. Il est possible d'exclure la possibilité de l'existence d'un arc quand, en présence d'une base limitée, les tests indiquent comme étant indépendantes deux variables pourtant corrélées dans le graphe d'origine.

Un autre façon d'appliquer ce principe est de limiter l'espace de recherche non plus à partir d'une procédure automatique mais à partir d'une connaissance *a priori*, généralement fournie par un expert [Acid et de Campos, 1996, de Campos et Castellano, 2007].

L'algorithme EGS (pour *Essential Graph Search*) de [Dash et Druzdzel, 1999] s'efforce quant à lui de construire le *GE* correspondant à la structure recherchée en évaluant par l'intermédiaire du score bayésien la solution renvoyée par des itérations successives de l'algorithme PC en faisant aléatoirement varier certains facteurs tels que le seuil de confiance des tests statistiques employés et un ordre topologique sur les variables du domaine.

4.5 L'apprentissage de la structure par des méthodes stochastiques

Jusqu'ici, les méthodes que nous avons présentées étaient déterministes, dans le sens où, à partir d'une même initialisation (par exemple, un même *GOSC* initial pour l'algorithme glouton) et d'une même base d'apprentissage, la solution retournée par ces algorithmes est la même.

Il existe une autre forme de recherche faisant appel à une part d'aléatoire : les algorithmes stochastiques. Ces méthodes font intervenir une part de hasard dans leur phase de recherche, ceci leur permettant notamment de se retrouver bloqués en certaines zones de l'espace des solutions.

Dans le cadre de l'apprentissage de structures de réseaux bayésiens, deux familles de méthodes stochastiques ont jusqu'à présent été employées : les méthodes dites de Monte-Carlo et les algorithmes évolutionnaires. Ces derniers faisant l'objet du prochain chapitre, nous nous contenterons de les évoquer brièvement à la fin de cette section (les applications des méthodes évolutionnaires à l'apprentissage de structures seront traitées à la fin du chapitre 5).

4.5.1 Méthodes de Monte Carlo par chaînes de Markov

Les méthodes de Monte Carlo forment un sujet trop vaste pour être traité de manière complète dans ce travail de thèse. Nous ne présentons ici que les éléments nécessaires à la compréhension générale de leur principe et de leur application à l'apprentissage de structures. Une très bonne introduction aux méthodes de Monte Carlo peut être trouvées dans [Robert et Casella, 2004] ; les méthodes de Monte Carlo par chaîne de Markov sont, quant à elles, décrites et expliquées dans [Gilks et al., 1996].

Les méthodes de Monte Carlo permettent l'approximation de distributions de probabilités à partir d'un échantillonnage (ou observations répétées des valeurs prises par la distribution de probabilités). Il existe de nombreuses approches suivant la connaissance *a priori* de la distribution étudiée et de la complexité même de l'échantillonnage. Nous nous intéressons particulièrement à une catégorie de méthodes : les méthodes de Monte Carlo par chaîne de Markov.

Ces méthodes modélisent une marche aléatoire dans l'espace de définition $\mathcal{X}$ d'une distribution de probabilités $\rho(x)$ et ce par le biais d'une chaîne de Markov. Cette chaîne sert alors de source d'échantillons de $\mathcal{X}$. Une chaîne de Markov est définie par un ensemble d'états, un ensemble des probabilités de transition entre ces différents états (aussi appelé *noyau* de la chaîne) et par une distribution de probabilités initiales (probabilités de se trouver en un état à l'initialisation).

Définition 23 (Chaîne de Markov) *Une chaîne de Markov définie sur un espace d'états $\mathcal{X}$, est un processus stochastique $(x^{(0)}, x^{(1)}, \ldots, x^{(t)})$ tel que :*

$$P(x^{(t)}|x^{(t-1)}, \ldots, x^{(0)}) = P(x^{(t)}|x^{(t-1)}), \forall t \in 1 \ldots T$$

Par la suite, nous employons la notation $x^{(t)}$ pour désigner indifféremment l'état de la chaîne de Markov au temps t et le t^e échantillon prélevé. Le noyau de la chaîne est un ensemble de probabilités de transition inter-états $K(x^{(t+1)}|x^{(t)})$ caractérisant la chaîne. Lorsque les probabilités de K sont constantes dans le temps (*i.e.* $K(x^{(t+1)} = x_i|x^{(t)} = x_j) = K(x_j|x_i)$, la chaîne est dite *homogène*). La chaîne est aussi définie par sa distribution de probabilité initiale $\rho_0(x^{(0)})$.

Distribution de probabilité stationnaire

On nomme *distribution stationnaire* d'une chaîne de Markov, distribution notée $\rho^*(x)$, une distribution de probabilités définie sur l'ensemble des états de la chaine et invariable dans le temps :

$$\rho^*(x) = \sum_{x'} K(x|x')\rho^*(x')$$

Cette convergence signifie qu'après un certain nombre, généralement élevé, de transitions, la distribution des observations des états parcourus par la chaîne tend vers une distribution fixe. Cette durée précédent la convergence est communément appelée période de *burn-in*. Sous certaines conditions (se référer à [Gilks et al., 1996] pour une description complète de ces

conditions), il est possible de garantir la convergence de la chaîne de Markov vers une telle distribution stationnaire $\rho^*(x)$.

L'objectif, dans le cadre d'une méthode MCMC, est alors que $\rho^*(x)$ approxime la distribution $\rho(x)$ étudiée. Des algorithmes, tels que l'algorithme de Metropolis-Hastings [Metropolis et al., 1953] ou l'échantillonneur de Gibbs [Geman et Geman, 1984], permettent d'y parvenir.

Méthodes MCMC sur l'espace des structures

[Madigan et York, 1995] proposent d'approximer la distribution *a priori* $P(\mathcal{G}|D)$ d'un *GOSC* $\mathcal{G}$ en connaissance de la base de données D. Pour y parvenir, l'échantillonnage direct sur l'espace des *GOSC* pourrait être envisagé mais, étant donné la taille de cet espace (cf. section 4.4), une approche par méthode MCMC est préférée.

Une chaîne de Markov est alors définie sur l'espace des *GOSC*. Les transitions inter-états de la chaîne, correspondant aux passages d'une structure à une autre, sont définies par l'application locale d'un opérateur d'ajout/soustraction/inversion d'un arc. Les probabilités de transition entre deux structures $\mathcal{G}$ et $\mathcal{G}'$ sont définies par une distribution de probabilités localement uniforme :

$$q(\mathcal{G}'|\mathcal{G}) = \frac{1}{|GOSC \text{ voisins de } \mathcal{G}|}$$

Cette méthode est bâtie sur le principe de l'algorithme de Métropolis-Hastings (décrit dans l'algorithme 4) et définit donc une probabilité d'acceptation d'un nouvel état. Cette probabilité est calculée à partir des probabilités *a posteriori* des modèles, probabilités qui sont, en pratique, calculées à partir d'une fonction de score (score bayésien, par exemple).

Algorithme 4 Algorithme de Metropolis-Hastings pour l'apprentissage de structures

Entrée: burn-in B, base d'apprentissage constituée de N cas
Sortie: $\mathcal{G}_{B+1}, \ldots \mathcal{G}_{B+N}$

$t \leftarrow 0$
$\mathcal{G} \leftarrow \mathcal{G}_0$, (aléatoire)
Pour t=1,. . . ,B+N **Faire**

- Proposer $\mathcal{G}'$, voisine de $\mathcal{G}_t$, avec la probabilité $q(\mathcal{G}'|\mathcal{G}_t)$
- Calculer R avec

$$R = min\left(1, \frac{P(\mathcal{G}'|D)q(\mathcal{G}'|\mathcal{G}_t)}{P(\mathcal{G}_t|D)q(\mathcal{G}_t|\mathcal{G}')}\right)$$

- Échantillonner u, variable uniforme sur (0,1)

Si u<R **Alors**
$\mathcal{G}_{t+1} \leftarrow \mathcal{G}'$
Sinon
$\mathcal{G}_{t+1} \leftarrow \mathcal{G}_t$
Fin Si
$t \leftarrow t + 1$
Fin Pour

Le quotient $\frac{P(\mathcal{G}'|D)}{P(\mathcal{G}|D)}$, aussi appelé *facteur de Bayes*, est calculé à partir d'un score bayésien tel que le score BD (cf. section 4.3).

Le *burn-in* et la qualité de la convergence de la chaîne ainsi définie sont fortement dépendants de la taille du problème (*i.e.* le nombre de variables du modèle) et il est fréquent de converger en un optimum local. Plusieurs lancements à partir de points distincts de l'espace échantillonné peuvent alors être effectués.

Méthodes MCMC sur l'espace des ordres topologiques

Une alternative, issue de [Friedman et Koller, 2000], consiste à parcourir l'espace des ordres topologiques (cf. définition 12). Ce type d'approche, aussi employé en conjugaison avec des algorithmes évolutionnaires (cf. section 5.5), combine une recherche sur un espace de taille bien plus restreinte (égale à $n!$) que celui des structures avec l'emploi d'une méthode renvoyant la meilleure structure pour un ordre donné (l'algorithme K2 – cf section 4.4.2). On peut évaluer un ordre $\prec$ donné en sommant sur l'ensemble des graphes topologiquement corrects avec celui-ci, de manière à obtenir $P(f|D)$. Cette somme est généralement approximée en ne considérant qu'un nombre restreint de structures (ayant, par exemple, un nombre maximal autorisé de parents par sommet dans le graphe).

Un mouvement au sein de l'espace des ordres topologiques correspond à une permutation de deux variables dans l'ordre topologique courant ou bien à une "coupe" de celui-ci (échange de deux parties). Cette approche est coûteuse pour chaque mouvement mais a lieu dans un espace beaucoup plus restreint que celui des structures.

Bien que cette approche s'avère plus performante qu'un échantillonnage dans l'espace des structures, [Eaton, 2007] soulève le problème de la détermination des distributions *a priori* sur les structures. Les probabilités *a priori* $P(\prec)$ des ordres topologiques, d'une part, et celles des structures pour un ordre topologique particulier $P(\mathcal{G}, \prec)$, d'autre part, sont uniformes. Cette uniformité résulte cependant en une distribution *a priori* $P(\mathcal{G})$ non uniforme : les structures en accord avec un nombre supérieur d'ordres (les structures les plus simples, telles que la structure vide $\mathcal{G}_0$), reçoivent une probabilité *a priori* $P(\mathcal{G})$ plus importante.

En sus des problèmes précédemment évoqués, les approches MCMC appliquées à l'apprentissage de structures présentent des inconvénients inhérents aux méthodes MCMC en général :

- le temps nécessaire à la convergence de la chaîne (le *burn-in*) peut être très longue. Un moyen de le réduire est d'injecter un certain nombre de contraintes. On peut ainsi définir un nombre de parents maximal par variable afin de réduire l'espace parcouru,
- la détection même de la convergence peut aussi poser problème. Aujourd'hui, cela reste un problème ouvert.

4.5.2 Méthodes évolutionnaires

Les méthodes évolutionnaires regroupent en leur sein de nombreux algorithmes, souvent distincts dans leurs fonctionnements, mais pouvant être ramenés au principe de l'optimisation de la solution à un problème à travers la simulation des préceptes de l'évolution biologique.

Nos travaux sont basés sur l'emploi de ces mêmes algorithmes évolutionnaires, et plus particulièrement les algorithmes génétiques. Le chapitre suivant introduit donc les différentes

catégories d'algorithmes évolutionnaires connus avant de présenter les applications existantes de ces méthodes à l'apprentissage de la structure d'un réseau bayésien.

4.6 Problématiques particulières

Les méthodes étudiées jusqu'à présent s'inscrivent dans une problématique similaire à celle de nos travaux, à savoir l'apprentissage de structures de réseaux bayésiens supposant :

- les variables du domaine sont discrètes ;
- les bases d'apprentissage utilisées sont complètes ;
- le problème est causalement suffisant.

Afin de proposer un panorama complet des méthodes d'apprentissage, cette section s'emploie à décrire les méthodes employées lorsque ces hypothèses ne sont pas vérifiées.

4.6.1 Cas des variables continues

Les méthodes d'apprentissage de structures peuvent elles aussi être adaptées au cas continus, soit par discrétisation de la base d'apprentissage afin d'apprendre un modèle discret, soit bénéficient elles aussi d'adaptation au domaine continu. [Colot et al., 1994] emploient une version modifiée du critère AIC afin de discrétiser l'espace des données. Cette pré-discrétisation, appliquée à la base d'apprentissage, servant à l'apprentissage d'un modèle lui-même discret. Les méthodes de discrétisation peuvent elles-mêmes faire partie intégrante de l'apprentissage ; [Friedman et Goldszmidt, 1996] fournissent une méthode de discrétisation à l'aide du critère MDL, le modèle appris et les données d'apprentissage discrétisées sont alors alternativement réévaluées afin d'optimiser leur adéquation mutuelle. D'autres méthodes ne discrétisent pas la base d'apprentissage ; ainsi [Margaritis, 2005] propose une méthode de test d'indépendance conditionnelle entre deux variables continues. Ce test pouvant alors être employé au sein d'un algorithme tel que l'algorithme PC (cf. section 4.2.1). Une description étendue des techniques d'apprentissage de structures dans le cas continu peut être trouvée dans [Fu, 2005].

4.6.2 Cas des bases de données incomplètes : l'algorithme SEM

À la différence des méthodes vues précédemment, l'algorithme SEM (*Structural Expectation Maximisation*) s'inscrit dans le cadre de la recherche du meilleur modèle dans le cas où la base d'apprentissage est incomplète.

Issu initialement des travaux de [Friedman, 1997], prolongés dans [Friedman, 1998], SEM conjugue la méthodologie d'estimation des données vue dans la section à l'apprentissage par évaluation dans l'espace des *GOSC* pour retourner simultanément la meilleure structure et les paramètres estimés associés.

Les méthodes de détermination de structures vues dans cette section calculent la vraisemblance du modèle candidat par rapport aux données en lui associant un score. Dans le cas jusqu'alors étudié, celui de données complètes, le calcul de cette vraisemblance revient à la

décomposer en un produit de termes liés au choix des parents des différentes variables du modèle ainsi qu'aux statistiques déterminées à partir de la base d'apprentissage.

SEM calcule la meilleure estimation des données non observées (et, par la suite, les paramètres calculés à partir de celles-ci) avant d'effectuer une recherche classique telle qu'en présence de données complètes.

L'algorithme 5 décrit le fonctionnement de SEM.

Algorithme 5 Algorithme Structural EM

1: Initialiser la structure et les paramètres associés $\mathcal{G}_0$,Θ_0
2: $t \leftarrow 0$ (itérations sur les structures)
3: **Tant que** $t \leq t_{max}$ **Faire**
4: **Tant que** $Score(\mathcal{G}_t) \geq Score(\mathcal{G}_{t-1})$ **Faire**
5: Phase d'espérance : estimation des paramètres Θ_t par EM paramétrique
6: **Tant que** $|\Theta^{t,m} - \Theta^{t,m-1}| \geq \epsilon$ ou $m < m_{max}$ **Faire**
7: • $N^*_{ijk} = E(N_{ijk} = \sum_{l=1}^{N} P(X_i = x_k | \Pi_i = \pi_{ij}, D_O^{(l)}, \Theta^{(t)})$
• $\theta^{t,m}_{ijk} = \frac{N^*_{ijk}}{\sum_{k=1}^{r_i} N^*_{ijk}}$
• $m \leftarrow m + 1$
8: **Fin Tant que**
9: Phase de maximisation : recherche de la structure $\mathcal{G}^{t+1}$
– création de l'ensemble $\mathcal{G}_\mathcal{V}$ des *GOSC* voisins de $\mathcal{G}_{t-1}$ à une opération d'inversion, soustraction ou addition d'un arc près,
– calcul de $Score(\mathcal{G}_v)$ pour chaque $\mathcal{G}_v \in \mathcal{G}_\mathcal{V}$,
– $t \leftarrow t + 1$
10: **Fin Tant que**
11: $\mathcal{G}_{t-1} \leftarrow \mathcal{G}_t$
12: **Fin Tant que**

A chaque itération sur t, l'algorithme SEM sélectionne la structure la mieux valuée et les paramètres associés. L'algorithme s'appuie sur le fait qu'il essaie d'améliorer le score à chacune de ses itérations. C'est là que se situe le point faible de SEM : pour l'algorithme EM standard, les points de convergence correspondent aux points où la fonction objectif est stationnaire ; mais cette notion n'est pas applicable dans l'espace des structures de modèles. Le problème survient quand l'algorithme converge vers un maximum local : ce cas se produit si un modèle génère une distribution assez performante pour faire apparaître les autres solutions comme étant moins performantes, au vu du score espéré.

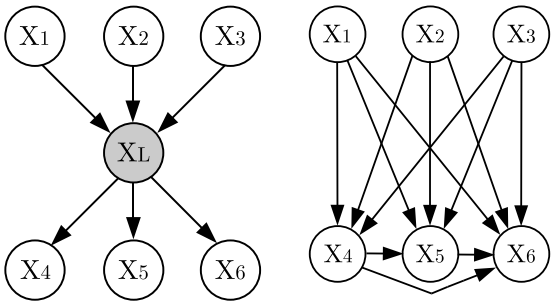
Plus la quantité d'information manquante est importante, plus cela risque de se produire.

Il existe plusieurs méthodes permettant l'apprentissage de structures à partir de bases de données incomplètes. Néanmoins, cette problématique ne faisant pas partie de notre champ d'études, nous invitons le lecteur qui souhaiterait en apprendre d'avantage sur le sujet à se reporter à [François, 2006], qui établit un panorama étendu et détaillé de ces différentes méthodologies ainsi qu'une description complète de la problématique.

4.6.3 Cas des variables latentes

Les algorithmes présentés dans ce travail de thèse, y compris nos propres travaux, font l'hypothèse de la suffisance causale (cf. section 4.1.1). Or, il se peut que dans le traitement d'un cas réel certaines variables observées aient une cause commune qui, elle, n'est pas observée ; une telle variable est alors appelée variable *latente*.

A priori, négliger une variable latente ne semble pas avoir de graves conséquences. Ceci est en partie vrai : l'apprentissage d'une structure ne tenant compte que des variables observées/connues permet théoriquement d'obtenir une I-map du modèle considéré, rassemblant l'ensemble des indépendances conditionnelles au sein de celui-ci. Le réel problème, en dehors de la véracité même de la modélisation est une complication du modèle obtenu.



(a) avec variable latente (b) sans variable latente

Figure 4.5 – Exemple de réseau appris en prenant en compte ou non une variable latente X_L.

Soit l'exemple de la figure 4.5(a), correspondant au *vrai* modèle que nous cherchons à apprendre. La variable V_L est une variable que nous considérons, dans la figure 4.5 comme latente. Le modèle obtenu en l'absence de cette variable, représenté sur la figure 4.5(b), présente un nombre élevé d'arcs comparativement au modèle complet de la figure 4.5(a). Des dépendances conditionnelles sont cependant bien détectées entre les variables $X_1, \ldots, X_6$. Le problème d'une telle modélisation est que, pour un nombre élevé de dépendances, la paramétrisation du modèle devient non seulement plus complexe (59 paramètres si les variables sont binaires contre 17 paramètres pour le modèle de la figure 4.5(a)) mais, consécutivement, la quantité de données nécessaire pour assurer la fiabilité des paramètres devient très élevée.

La problématique des variables latentes ne fait pas partie de ce travail de thèse et nécessite un ensemble complet de notions, de définitions et d'hypothèses qu'il serait impossible d'énumérer ici. Par conséquent, cette section se limitera donc à une présentation résumée des principales notions et travaux nécessaires à la compréhension du problème.

De la même manière que pour les méthodes appliquées aux problèmes causalement suffisants, les méthodes prenant en charge les problèmes à variables latentes peuvent se répartir en méthodes à tests statistiques et méthodes employant un score.

Méthodes statistiques

Les auteurs de l'algorithme PC ont développé une variante de celui-ci prenant en compte les variables latentes : l'algorithme FCI (pour *Fast Causal Inference*) [Spirtes, 2001, Spirtes et al., 2000].

En parallèle, les auteurs de l'algorithme IC ont, eux aussi, développé une extension de ce dernier, nommée *IC** [Pearl, 2000].

Ces algorithmes ne renvoient pas un *GOSC*, mais un *graphe complet partiellement ancestral* (ou *PAG*, pour *Partial Ancestral Graph*). Un *PAG* permet de représenter, pour une même classe d'équivalence au sens de Markov, l'ensemble des indépendances conditionnelles du modèle (ce qu'un *GOSC* n'est pas toujours capable de faire). Nous limitons par la suite notre description des *PAG* au traitement des modèles à variables latentes ; pour une description complète, le lecteur se reportera à [Spirtes et al., 2000].

Dans ce qui suit, nous considérons le sous ensemble O de l'ensemble des variables $\mathcal{V}$ comme représentant les variables *observées* (*i.e.* connues) et *Cond*, l'ensemble des indépendances conditionnelles existant au sein des variables de O.

Soit $Eq(\mathcal{G}, O)$ la classe d'équivalence au sens de Markov de l'ensemble des *GOSC* $\mathcal{G}'$ tels que $\mathcal{G}'$ soit défini sur un super-ensemble de O et tels que la condition de Markov globale affirme les indépendances conditionnelles de *Cond*.

Définition 24 *ψ est un PAG pour le GOSC $\mathcal{G}$ doté de l'ensemble de sommets $\mathcal{V}$ et de l'ensemble de variables observées $O \subseteq \mathcal{V}$ si et seulement si :*

1. *ψ est défini sur O*
2. *Il existe une arête $X -> Y$ dans ψ si et seulement si X cause Y*
3. *S'il existe une arête $X \leftrightarrow Y$ dans ψ alors dans tout graphe de $Eq(\mathcal{G}, O)$, X et Y sont les conséquences d'une troisième variable Z, latente*
4. *S'il existe une arête $X \circ\!\!-\!\!> Y$ dans ψ alors, dans tout graphe de $Eq(\mathcal{G}, O)$, Y n'est pas un ancêtre de X : soit $X \rightarrow Y$, soit $X \leftrightarrow Y$*
5. *Une arête $X \circ\!\!-\!\!\circ Y$ ne permet aucune conclusion quant aux relations de parenté entre X et Y dans $Eq(\mathcal{G}, O)$ ou quant à l'existence d'une variable latente qui serait leur cause commune*

La détermination du *PAG*, pour l'algorithme IC*, se fait en plusieurs temps. Dans un premier temps, un graphe non-orienté est déterminé par une suite de tests d'indépendance conditionnelle. Puis, la détection de V-structures permet une première orientation du graphe obtenu précédemment. Ensuite, une série de règles (non décrites ici mais définies dans [Pearl, 2000]) permet la détermination des arêtes du *PAG* final.

Les règles de construction du *PAG* mise en place dans l'algorithme FCI ont cependant été démontrées comme étant incomplètes et l'algorithme s'est vu augmenté et complété par J. Zhang [Zhang, 2006]. Bien que ces méthodes reposent sur la notion de causalité, les arcs des solutions renvoyées ne tiennent plus nécessairement compte de cette dernière notion (un modèle causal peut cependant être obtenu à partir d'un *PAG* [Meganck et al., 2007]).

Enfin une approche similaire (basée sur une lecture de la structure sur les variables observées) a été proposée par [Elidan, 2004] et consiste à détecter des "signatures graphiques" au sein du graphe formé sur les variables observées et d'évaluer les structures augmentées par l'intermédiaire d'une version adaptée de l'algorithme EM.

Algorithme 6 Algorithme IC*

Notations : $X_i - *X_j$ signifie $X_i - X_j$ ou $X_i \rightarrow X_j$ ou $X_i \multimap X_j$

1^e ***étape : Construction d'un graphe non-orienté***
Soit $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, le graphe ne relient aucun sommet de $\mathcal{V}$
Pour chaque couple de variables $(X_i, X_j) \in \mathcal{V}^2$ **Faire**
  Chercher $SepSet_{\mathcal{G}}(X_i, X_j)$ tel que $X_i \perp X_j | SepSet_{\mathcal{G}}(X_i, X_j)$
  Si $SepSet_{\mathcal{G}}(X_i, X_j) = \emptyset$ alors ajouter l'arête $X_i \circ\!\!-\!\!\circ X_j$ à $\mathcal{G}$
Fin Pour

2^e ***étape : détection des V-structures***
Pour Chaque triplet de $\mathcal{V}^3$ tel que $X_i * - * Z, Z * - * X_j \in \mathcal{E}^2, X_i * - * X_j \notin \mathcal{E}$ **Faire**
  Si $Z \notin SepSet_{\mathcal{G}}(X_i, X_j)$ **Alors**
    Orienter : $X_i* \rightarrow Z \leftarrow *X_j$ dans $\mathcal{G}$
  Fin Si
Fin Pour

3^e ***étape : orientation des arêtes restantes***
Répéter
$\forall (X_i, X_j) \in \mathcal{V}^2$
Si $X_i * - * X_j$ et $\exists$ un chemin orienté de X_i vers X_j **Alors**
  orienter l'arête $X_i * - * X_j$ en $X_i* \rightarrow X_j$
Sinon
  Si $X_j \notin Adj_{\mathcal{G}}(X_i), \forall Z$ tel que $X_i* \rightarrow Z$ et $Z * - * X_j$ **Alors**
    orienter l'arête $Z * - * X_j$ en $Z \rightarrow X_j$
  Fin Si
Fin Si
Tant qu'il est possible d'orienter des arêtes dans $\mathcal{G}$

Méthodes employant un score

L'algorithme SEM (cf. section 4.6.2) peut être adapté pour la détection de variables latentes en considérant celles-ci comme des variables dont la totalité des données est manquante lors de l'étape E de l'algorithme. Cependant, SEM prend en entrée le nombre de variables latentes ainsi que leurs cardinalités respectives.

[Zhang, 2003] a adapté l'algorithme SEM afin de permettre la détection de variables latentes auprès de modèles hiérarchiques latents – une forme particulière de modèle limité à une arborescence et dont seules les feuilles sont observées –.

Un autre problème, commun à l'ensemble des méthodes, est la détermination de la cardinalité d'une variable latente : conférer une cardinalité trop faible pour une variable X revient dans la plupart des cas à compliquer la structure du modèle en reliant les variables enfants de X entre elles afin de mieux décrire le domaine. [Elidan et Friedman, 2001] introduisent ainsi une méthode revenant à effectuer une sélection de modèles en faisant varier les différentes cardinalités envisageables.

Chapitre 5

Algorithmes génétiques

Le terme de méthodes évolutionnaires regroupe, de nos jours, plusieurs méthodologies différentes ayant pour point commun de s'inspirer, du moins à l'origine, des théories darwiniennes de l'évolution. Ces méthodes proposent ainsi d'améliorer un ensemble de solutions en simulant une succession de générations au cours desquelles ces solutions subissent une pression visant à favoriser la survie des meilleures d'entre elles et sont modifiées par des opérateurs dédiés, un principe évoqué dès le milieu du XX^e siècle par Alan Turing [Turing, 1948].

Ce chapitre a pour objectif d'introduire les principales notions liées à ce domaine et plus particulièrement aux algorithmes génétiques afin d'aider à la compréhension des méthodes que nous avons développées.

L'organisation de ce chapitre est la suivante :

1. nous commencerons par présenter les principes généraux des méthodes évolutionnaires,
2. nous ferons par la suite une description générale d'un algorithme génétique,
3. nous aborderons plus en détail les composantes d'un algorithme génétique,
4. l'aspect historique ainsi que la théorie fondatrice des algorithmes génétiques seront étudiés,
5. certains des principaux développements des algorithmes génétiques seront présentés.

Au préalable et dans un souci de clarté, nous souhaitons expliciter certains termes empruntés au vocabulaire des biologistes et couramment employés dans le cadre de l'algorithmique évolutionnaire.

individu : une solution candidate au problème considéré, souvent plusieurs solutions sont simultanément évoluées et constituent une *population* d'individus,

chromosome : la représentation d'un individu, plus exactement le codage de celui-ci au sein de l'algorithme génétique,

génération : un algorithme évolutionnaire itère à plusieurs reprises les mêmes opérations sur les individus, générant à chaque fois un nouveau lot d'individus pour l'itération suivante. Symboliquement, on désigne par *génération* chacune de ces itérations,

parents/enfants : les relations de paternité expriment communément un lien entre deux solutions candidates. Un enfant désignant une solution générée à partir d'une solution préexistante qui en est alors le parent,

génotype/phénotype : ces termes, hérités de la génétique, désignent respectivement l'encodage de la solution (le chromosome et les allèles de l'individu) et l'expression de celui-ci. Le phénotype est exprimé par une fonction f évaluant la qualité de la solution encodée par le génotype.

5.1 Introduction

Les algorithmes évolutionnaires sont une famille de méthodes de résolution de problèmes d'optimisation consistant à perfectionner un groupement de solutions candidates à la manière d'une population d'organismes vivants évoluant dans un écosystème.

Les différentes méthodes évolutionnaires se distinguent sur des points tels que l'aspect de l'évolution abordée (intra ou inter-espèces, par exemple), le codage des solutions ou, plus simplement, les problématiques auxquelles elles sont adaptées.

Stratégies d'évolution : ces méthodes (désignées par ES pour *evolutionary strategies*) firent leur apparition en 1965 [Rechenberg, 1970]. Le ES manipulent des vecteurs de nombres réels. Si les premiers ES faisaient évoluer un individu, les ES plus récents consistent en des stratégies, notées (μ, λ), où μ parents génèrent λ enfants[1]. Si les premiers algorithmes de type ES n'employaient pas d'opérateurs de recombinaison, des versions plus récentes font éventuellement évoluer plusieurs parents et procèdent à des opérations de croisement ;

Programmation évolutionnaire : apparues au milieu des années soixante [Fogel et al., 1966] et très proches des stratégies d'évolution, les méthodes de programmation évolutionnaire (ou EP pour *evolutionary programming*) considèrent les solutions évoluées comme autant d'espèces différentes et ne comporte donc pas de croisement – une exception qui est permise dans le cas des ES –. La sélection des meilleures solutions est stochastique alors que ce même processus est déterministe dans les ES,

Algorithmes génétiques : abordés en détail dans ce travail de thèse, les algorithmes génétiques sont issus des travaux de Holland [Holland, 1975] et se démarquent des méthodes précédentes sur des points tels que le recours au croisement,

Programmation génétique : la programmation génétique [Koza, 1989, Koza, 1992] est la plus récente des méthodes évolutionnaires. Elle est essentiellement dédiée à l'évolution de programmes ou d'autres expressions syntaxiques, représentés sous forme d'arbres et évolués suivant un principe similaire à celui des algorithmes génétiques.

Nos travaux et, par conséquent, les théories et résultats nous intéressant, sont exclusivement consacrés aux algorithmes génétiques. Le lecteur intéressé par les méthodes évolutionnaires en général pourra cependant trouver des études complètes sur celles-ci en se rapportant à des ouvrages et études traitant le sujet plus exhaustivement : [Eiben et Smith, 2003, De Jong, 2001,

[1] Nous employons exceptionnellement ici la notation λ par convention vis-à-vis de la littérature consacrée aux ES ; cependant cette notation λ, dans nos travaux, désigne la taille totale de la population.

Kallel et al., 2001]. Une introduction complémentaire aux algorithmes génétiques peut de même être trouvée dans [Whitley, 1994].

Nous portons maintenant notre attention sur les algorithmes génétiques à travers une présentation de ces algorithmes, une étude de leur implémentation d'origine ainsi que de leurs différentes composantes, pratiques et théoriques.

5.2 Les algorithmes génétiques

Les algorithmes génétiques ont connu, depuis leur création, des implémentations diverses ; néanmoins, la plupart de ces implémentations suivent le schéma du premier algorithme génétique proposé par Holland en 1975. Ce schéma est connu sous le nom d'*algorithme génétique canonique* et est décrit par l'algorithme 7.

Algorithme 7 Algorithme génétique canonique

*/ *Initialisation* /*
$t \leftarrow 0$;
Générer aléatoirement et uniformément une population initiale P_0 de λ individus et évaluer ceux-ci à l'aide d'une fonction f,
*/ *Evolution* /*

1. sélectionner des individus de P_t pour la reproduction ;
2. obtenir de nouveaux individus par application de l'opérateur de croisement sur les individus préalablement sélectionnés ;
3. appliquer un opérateur de mutation sur les nouveaux individus : les individus obtenus constituent la nouvelle population P_{t+1} ;

*/ *Évaluation* /*
Evaluer les individus de P_{t+1}
$t \leftarrow t + 1$;
*/ *Arrêt* /*
Si un critère défini est rencontré, arrêt, sinon, relancer la phase d'évolution

L'algorithme génétique canonique suit lui-même le schéma général de fonctionnement d'un algorithme évolutionnaire, résumé par la figure 5.1.

Un algorithme génétique consiste en une série d'opérations de manipulation et de sélection itérées sur une population d'individus. Nous allons, dans la suite, nous employer à décrire les différents aspects, conceptuels et pratiques, nécessaires à la construction d'un algorithme génétique.

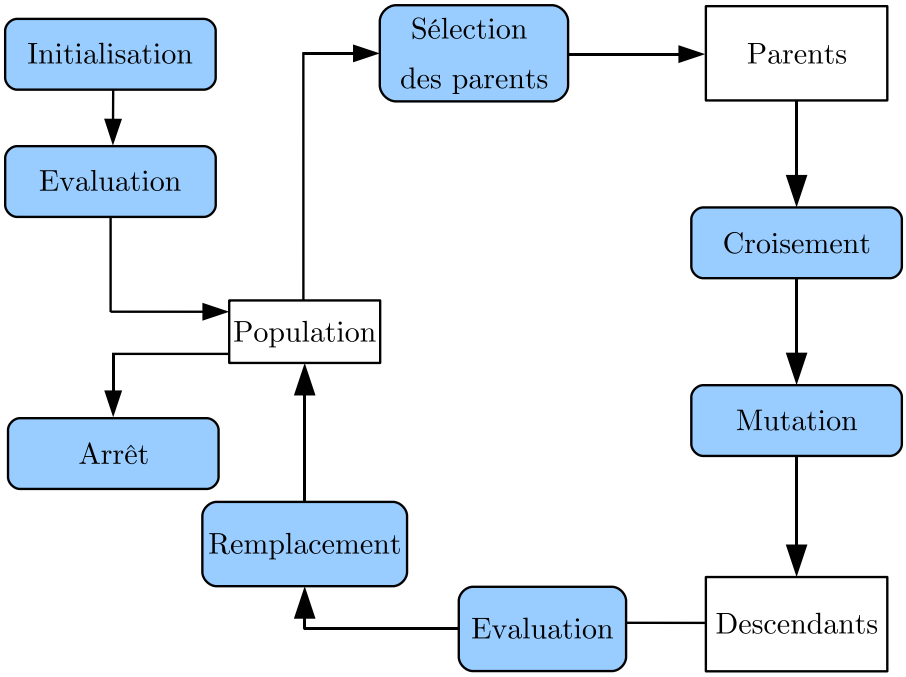


Figure 5.1 – Schéma général de fonctionnement d'un algorithme génétique.

5.2.1 Les composantes d'un algorithme génétique

Représentation des individus Comme nous l'avons précisé en entrée de chapitre, une solution en tant qu'individu se présente sous deux formes : son phénotype, qui est une mesure qualitative de la solution, et le génotype, qui est l'encodage de la solution.

La forme de l'encodage est primordial car il détermine entre autres l'efficacité avec laquelle vont pouvoir agir les différents opérateurs d'exploration et d'exploitation de l'espace des solutions. Alors que les premiers algorithmes génétiques travaillaient exclusivement sur l'espace des chaînes binaires (la justification de cette approche est abordée en section 5.3.1). Le problème du choix de la représentation dépend essentiellement de l'existence d'opérateurs adéquats.

Ainsi, depuis le premier algorithme travaillant sur les chaînes binaires, les espaces pouvant être utilisés au sein d'un algorithme génétique se sont diversifiés. Il est, de fait, possible de travailler sur n'importe quel espace de solutions tant que les opérateurs génotypiques présentent certaines propriétés [Radcliffe, 1991].

- possibilité d'explorer l'ensemble de l'espace des solutions en un nombre fini d'opérations (intérêt de la mutation) ;
- transmission des propriétés communes des parents à leurs enfants ;
- stabilité de la fonction d'évaluation au voisinage des individus (principe aussi appelé *strong causality principle* [Rechenberg, 1970]).

Des opérateurs adéquats ont ainsi été définis afin de travailler dans l'espace des réels, des permutations d'entiers ou encore des arbres.

Évaluation Un algorithme génétique est basé sur la performance des individus composant la population de solutions. Cette performance – le phénotype de l'individu concerné – est évaluée au travers d'une fonction : la *fitness*.

La performance peut revêtir plusieurs formes, il peut s'agir d'une concordance par rapport à une base de données, d'une valeur moyenne, etc. La fonction d'évaluation va prendre la forme d'une valeur que nous allons chercher à maximiser (ou, inversement, à minimiser) à travers l'évolution de la population.

Le choix de la fonction d'évaluation, bien qu'il soit fréquemment imposé par le contexte du problème, requiert une grande attention car le coût du calcul de cette fonction représente souvent la majeure partie du coût de l'algorithme génétique lui-même.

Initialisation Une fois que l'on a choisi la représentation et la fonction d'évaluation, la première étape préalable à la mise en marche du processus d'évolution consiste à initialiser une population de départ.

L'option la plus simple consiste fréquemment en une initialisation aléatoire, en plusieurs points de l'espace des solutions. Il est néanmoins souvent possible de tenter de créer une population d'ores et déjà en possession d'un génome "performant" et ce en faisant appel à une heuristique existante offrant des solutions sinon performantes, du moins de bonne qualité.

L'initialisation par une heuristique doit cependant être abordée avec précaution. Il est possible d'introduire un biais dans l'initialisation et de risquer une convergence prématurée vers un optimum local.

5.2.2 Opérateurs phénotypiques

En accord avec les principes darwiniens, les algorithmes génétiques entreprennent de favoriser l'apparition et la survie des individus les plus aptes. Cette pression s'exerce à partir la mesure de qualité ou *fitness* de ceux-ci.

Stratégies de sélection La sélection intervient lors de deux phases distinctes : la sélection d'individus pour le croisement et le remplacement des individus d'une génération à une autre.

Il est nécessaire, à chaque génération, de sélectionner des individus devant participer à la reproduction. Dans un souci de conservation de la diversité du matériel génétique à disposition, les opérateurs de croisement fonctionnent sur la base de probabilités. Ces probabilités favorisent les individus les plus aptes mais n'excluent pas la sélection d'individus moins performants et ceci afin de pouvoir garantir la diversité du matériel génétique de la population.

Les premières stratégies de sélection étaient directement proportionnelles à la *fitness* des individus considérés (on parle alors de sélection par roulette). Bien que reflétant l'aspiration première du principe de sélection, une telle approche risque de grandement favoriser d'éventuels super-individus (présentant une *fitness* bien supérieure à la moyenne) et de provoquer une convergence prématurée de la population. Par la suite, d'autres stratégies ont été développées telles que la sélection par tournoi, opposant des lots d'individus afin d'en sélectionner le

meilleur avec une certaine probabilité, ou bien encore la sélection par rang où la probabilité de sélection est proportionnelle au rang de la *fitness* de l'individu dans la population.

La principale différence entre les phases de sélection pour le remplacement et de sélection pour le croisement est qu'un individu peut être sélectionné, par tirage au sort, à plusieurs reprises pour participer au croisement. Lors de la phase de remplacement, un individu voit son sort décidé une fois pour toutes (survie ou disparition).

L'algorithme génétique canonique constitue chaque nouvelle population à partir des individus nouvellement créés ; cette approche risque cependant de poser un problème si un chromosome de très bonne qualité se voit altéré à la suite d'un croisement et/ou d'une mutation. La politique de renouvellement total de l'algorithme génétique canonique risque alors d'entraîner une dégradation de la meilleure solution jusqu'alors rencontrée. [Rudolph, 1994] montre que l'algorithme génétique canonique ne saurait garantir la convergence vers un optimum global sans conservation d'une génération à l'autre de la meilleure solution trouvée.

On parle de stratégie *élitiste* lorsque le meilleur enfant est moins performant que le meilleur parent, celui-ci est automatiquement transmis à la génération suivante. Cette conservation d'une génération à l'autre est en particulier souhaitable lorsque le coût du calcul de la *fitness* d'un individu est élevé. Le revers étant que l'on risque, en particulier avec la conservation d'un trop grand nombre d'individus, une convergence prématurée de la population en un optimum local.

5.2.3 Opérateurs génotypiques

Ces opérateurs travaillent directement sur la représentation (le chromosome) de l'individu afin de générer de nouvelles solutions par recombinaison avec d'autres éléments de solution ou bien le modifiant aléatoirement.

Opérateur de croisement L'opérateur de croisement prend en entrée deux (ou plus) parents et recombine différents éléments issus de ceux-ci afin de générer un ou plusieurs individus.

Lorsque les individus croisés sont génétiquement proches (leurs génotypes diffèrent peu), le croisement a une vocation d'exploitation de l'existant. Ce rôle est cependant modifié lorsque les parents sont très différents, dans ce cas l'opérateur devient un opérateur d'exploration de l'espace.

Alors que l'algorithme génétique canonique effectue le croisement par échange simple des moitiés de chromosomes des parents, d'autres types d'opérateurs de croisement sont apparus. Les possibilités actuelles sont nombreuses et dépendent de la problématique abordée : opérateurs de croisements en plusieurs points, fusion de solutions, croisement linéaire avec des poids tirés aléatoirement, etc.

Opérateur de mutation Par analogie avec les théories darwiniennes selon lesquelles le génome d'une population évolue ponctuellement par l'intermédiaire de mutations localisées, l'opérateur de mutation d'un algorithme génétique a pour rôle l'exploration de l'espace des

solutions du problème considéré par une série de perturbations aléatoires des chromosomes de la population.

Sur une représentation telle qu'une chaîne de bits, cette modification consiste, par exemple, à inverser aléatoirement un ou plusieurs bits.

Le fonctionnement de l'opérateur de mutation est stochastique : une mutation peut survenir en chaque gène ou ponctuellement en un gène aléatoirement choisi d'un individu avec une probabilité P_{mute}.

Critère d'arrêt On peut décider de la terminaison des calculs après un certain délai (temps ou nombre d'itérations), après que le meilleur individu ait atteint un certain niveau de qualité, après un certain nombre d'itérations sans amélioration du meilleur individu ou bien lorsque l'écart type de la *fitness* de la population passe en deçà d'un certain seuil.

Les paramètres d'un algorithme génétique En dehors des différents opérateurs et stratégies entrant en compte dans l'implémentation d'un algorithme génétique, il existe un certain nombre de paramètres à définir :

Taille de la population : Si de manière intuitive, une population de grande taille paraît être une solution idéale et ce afin de mieux explorer l'espace de recherche, l'augmentation en terme de coût de calculs supplémentaires n'est pas à négliger. En l'absence d'une stratégie visant à adapter ce paramètre au long du fonctionnement de l'algorithme (voir [Eiben et al., 2004, Eiben et al., 2006]pour des exemples de stratégies d'adaptation de la taille de la population), le retour d'expérience reste le manière la plus répandue de fixer ce paramètre ;

Probabilité de croisement : Une fois qu'un ensemble d'individus a été sélectionné pour participer à la reproduction, ceux-ci n'y participeront réellement qu'avec une probabilité dite de croisement, P_{cross} ;

Probabilité de mutation : Sujet de nombreuses discussions, la probabilité de mutation P_{mute} représente la probabilité avec laquelle un individu (ou un de ses gènes) subira l'effet de l'opérateur de mutation. Si une probabilité élevée permet une grande couverture de l'espace de recherche, une solution proche de l'optimal recherché n'aura alors que peu de chances d'y parvenir suite à une perturbation trop forte. Inversement, une probabilité trop faible restreint la recherche et diminue d'autant les chances d'approcher cet optimal ;

Le paramétrage d'un algorithme génétique est délicat. Souvent, le paramétrage est établi soit à partir d'une connaissance *a priori* du problème à résoudre, soit, le plus souvent, à partir d'un plan d'expérience et de nombreux essais. Nous verrons que certaines recherches ont visé à automatiser le paramétrage en laissant ce dernier, au même titre que la population même, à la discrétion du processus évolutionnaire.

Il faut rappeler qu'il n'existe pas de paramétrage universellement performant pour un algorithme génétique. S'il existe des valeurs acceptées pour certains paramètres – on emploie souvent une probabilité de croisement proche des 0,80 ainsi qu'une probabilité de mutation en $O(\frac{1}{n})$ avec n, le nombre de gènes du chromosome – il est presque toujours nécessaire de passer par une phase d'étalonnage.

5.2.4 Applications à des problèmes continus

Les stratégies d'évolution, comme nous l'avons mentionné précédemment, ont été conçus pour la manipulation de vecteurs de nombres réels ; mais les algorithmes génétiques peuvent, eux aussi, être employés pour la résolution de problèmes d'optimisation continue – problèmes pour lesquels la fonction à optimiser est définie sur l'espace des réels –.

À l'origine, les algorithmes génétiques étaient conçus pour ne manipuler que des chaînes binaires et les premières approches des problèmes continus consistaient en un codage des valeurs réelles sous forme de chaînes binaires. Cette approche a pour inconvénient, d'une part, un manque de précision dans le codage des solutions (plus on doit être précis, plus une chaîne binaire encodant un nombre réel doit être longue) et, d'autre part, pose le problème d'incongruité entre une distance dans l'espace des génotypes (binaires), la distance de Hamming, par exemple, et une distance dans l'espace des phénotypes (réels) [Deb et Agrawal, 1995]. Bien que d'autres mesures dans l'espace binaire permettent de s'affranchir, en partie, du deuxième point soulevé (e.g. l'utilisation d'un codage de Gray), la dernière décennie a vu apparaître un nombre croissant de travaux s'appliquant à travailler directement avec des représentations réelles, à l'aide d'opérateurs *ad hoc* [Davis, 1991, Parker, 2002] ou d'autres représentations intermédiaires [Surry et Radcliffe, 1997].

Des compléments quant à l'optimisation continue par des méthodes évolutionnaires peuvent être trouvés dans [Auger, 2004]. Nous verrons, plus loin dans ce chapitre, que d'autres types de méthodes évolutionnaires – les algorithmes à estimation de distribution – peuvent, eux aussi, travailler sur des espaces continus.

Si cette section s'est attachée à présenter les éléments constitutifs couramment implémentés dans un algorithme génétique, nous allons aborder dans la suite un rapide descriptif des théories et concepts sur lesquels se basent les algorithmes génétiques.

5.3 Étude théorique

Cette section a pour objectif de présenter au lecteur les premières théories à la base des algorithmes génétiques ainsi que les dernières opinions et critiques vis-à-vis de celles-ci.

5.3.1 Le théorème des schémas

Quand Holland conçut les algorithmes génétiques, ce fut avant tout dans l'esprit d'une modélisation informatique du processus d'évolution tel que décrit par Darwin. Cependant, le principe mathématique initial des algorithmes génétiques était fondé sur le théorème des schémas, une théorie visant à expliquer le comportement d'un algorithme génétique en tant qu'optimiseur.

Un schéma est une chaîne formées d'éléments d'un alphabet $\mathcal{A}$ auxquels s'ajoute un terme $*$ employé comme "joker". Par exemple, si $\mathcal{A} = \{0, 1\}$, les chaînes {01011001} et {11000010} font

partie du schéma $H_1 = \{*10**0**\}$. Dans la suite de cette section, nous considérons que l'alphabet employé est $\mathcal{A} = \{0, 1\}$.

Un schéma est notamment caractérisé par son ordre, noté *ordre*(H), c'est à dire l'ensemble de ses éléments prenant une valeur fixe ($\in \{0, 1\}$). Dans l'exemple précédent, H_1, *ordre*(H_1) vaut 3. La longueur de définition d'un schéma est, elle, égale à l'écart maximal entre deux bits à valeurs fixes dans le schéma. Par exemple, $d(H_1) = 4$.

Le théorème des schémas vise à mesurer l'espérance du nombre d'occurrences $N(H, t + 1)$ d'un schéma H présent dans la population à un temps $t + 1$ en fonction du nombre de ces occurrences au temps t précédent, des caractéristiques du schéma et de la qualité des individus contenant ce schéma. Les calculs suivants s'appliquent dans le cadre du fonctionnement de l'algorithme génétique canonique, tel que défini dans la figure 5.1.

Cette espérance dépend de la persistance du schéma vis-à-vis des trois opérateurs de l'algorithme génétique (sélection, croisement, mutation) :

sélection Soit $f_t(H)$, la *fitness* du schéma H. Elle est calculée en moyennant la *fitness* de l'ensemble des membres du schéma présents dans la population au temps t. La probabilité de sélection est directement proportionnelle à la *fitness*, donc :

$$P_{select}(\text{une instance de } H) = \frac{f_t(H)}{\overline{f}_t}$$

Ce calcul ignore cependant les effets des opérateurs de croisement et de mutation, lesquels peuvent eux-mêmes affecter le schéma H,

croisement le schéma H est "coupé" par le croisement avec une probabilité égale à :

$$P_{cross} \times \frac{d(H)}{l-1}$$

où l représente la longueur du chromosome parent.
La probabilité de destruction du schéma est en réalité inférieure au terme précédent : si deux individus présentant le schéma H se croisent, H ne sera pas détruit. D'autre part, ce calcul ignore le cas où le croisement génère un individu appartenant à H ;

mutation la probabilité de ne pas détruire le schéma est égale à la probabilité de ne pas altérer au moins un de ses points fixes, au nombre de *ordre*(H), soit égale à : $1 - (P_{mute})^{ordre(H)}$.

En connaissance de ces trois facteurs, le théorème des schémas permet alors d'écrire l'inégalité suivante :

$$E\left[N(H, t+1)\right] \geq N(H, t) \times \frac{f_t(H)}{\overline{f}_t} \times \left(1 - P_{cross}.\frac{d(H)}{l-1}\right) \times (1 - P_{mute})^{ordre(H)} \tag{5.1}$$

En parallèle au théorème des schémas, deux notions virent le jour :

***Building blocks* :** Une hypothèse corrélée au théorème des schémas est l'hypothèse des blocs de construction – ou *building blocks* – développée par Goldberg [Goldberg, 1989]. La représentation d'un schéma H au sein de la population croît rapidement si H est très court (et peu susceptible d'être détruit par croisement ou mutation) et doté d'un ratio de *fitness* supérieur à 1. L'algorithme favorise alors la découverte et la recombinaison de ces blocs élémentaires de solutions,

Alphabets employés : les alphabets de faible cardinalité, employés pour le codage du génotype des individus évolués, devraient permettre l'échantillonnage, pour un chromosome assez long, d'un nombre maximum de schémas pour un seul individu.

Pendant longtemps, le théorème des schémas, l'emploi de l'alphabet binaire et la recherche des meilleurs blocs de construction guidèrent la conception des algorithmes génétiques. Comme nous allons le voir par la suite, ces différents points se sont cependant vus remis en question.

5.3.2 Critiques

Dès la fin des années quatre-vingts, le théorème des schémas fut sérieusement remis en question. Des lacunes dans la formulation du théorème ainsi que les observations des performances de l'algorithme génétique canonique sur diverses classes de problèmes mirent en évidence plusieurs problèmes, parmi lesquels :

- les alphabets de faible cardinalité ne permettent pas systématiquement un échantillonnage optimal de l'espace des schémas [Antonisse, 1989]. Une démonstration du raisonnement original de Holland ainsi que de celui d'Antonisse peut être trouvée dans [Koehler, 1997],
- La majorité des problèmes d'optimisation difficiles sont *trompeurs* [Whitley, 1991]. C'est-à-dire que les blocs de construction repérés comme performants par la fonction d'évaluation de l'algorithme ne font pas nécessairement partie de l'optimum recherché. Dans ce cas, la majorité des algorithmes génétiques ne trouvent pas l'optimum global recherché,
- la prolifération d'un schéma de bonne qualité, suggérée par le théorème des schémas, ignore complètement les autres schémas présents dans la population. À mesure que celle-ci converge, le ratio de la *fitness* du schéma sur la *fitness* moyenne converge vers 1 et le nombre de représentants se verra affecté par les opérations de croisement et de mutation,
- un autre point crucial ignoré dans l'énonciation du théorème est le choix de la représentation. Un choix *a priori* arbitraire de cette représentation interdit la prévalence de l'algorithme génétique car il est alors impossible de garantir la transmission de l'information sous forme de schémas [Radcliffe, 1992],
- comme nous l'avons mentionné, le théorème des schémas ignore les possibilités de génération de schémas par le biais du croisement.

Pour ces raisons, entre autres, il s'est vite avéré que le théorème des schémas ne permettait pas de justifier les performances de l'algorithme génétique et ne pouvait donc servir de base à l'élaboration d'un algorithme performant. Le théorème des schémas n'a plus aujourd'hui qu'une valeur historique.

S'il est vrai que la simulation du milieu naturel en tant que solveur de problème pouvait, dans un premier temps, laisser espérer l'obtention d'un solveur "universel", cet espoir a depuis été sérieusement remis en question.

Il a d'abord été prouvé que les algorithmes génétiques, sous leur forme originelle, n'étaient pas des optimiseurs [De Jong, 1992]. Principe conforté, par la suite, par le théorème du *No free lunch* (en français : "pas de repas gratuit") [Wolpert et Macready, 1995]. Ce théorème, qui n'en est pas vraiment un, statue que tout algorithme se comporte, en moyenne, de la même manière sur l'ensemble des problèmes d'optimisation ; si un algorithme est avantageux sur un problème particulier, il est inversement désavantagé sur les autres problèmes. En résumé, il ne peut exister

un algorithme universel pouvant résoudre l'ensemble des problèmes d'optimisation. Pour une classe de problèmes donnée, il est nécessaire de rendre l'algorithme optimal pour la résolution de cette classe, précisément.

Un tournant dans l'avancement des algorithmes génétiques furent les travaux de D. Davis [Davis, 1991].

Celui-ci a abordé les algorithmes évolutionnaires d'un point de vue purement pratique. Avant même que n'apparaisse le théorème du *no free lunch*, Davis statua le premier sur la nécessité de concevoir un algorithme génétique dédié et conçu autour du problème à résoudre. Davis mit en avant une approche "ingénieur" de l'algorithmique évolutionnaire. Les résultats obtenus par Davis eurent clairement une influence sur le devenir des algorithmes évolutionnaires. Les études circulant autour des algorithmes se sont rapidement divisées en deux branches : une première se proposait d'étudier ces algorithmes d'un point de vue pragmatique, en étudiant principalement les applications diverses qui pouvaient être faites dans le domaine de l'optimisation tandis qu'une deuxième branche s'intéressait plus particulièrement à la modélisation mathématique et à l'approche biomimétique des algorithmes évolutionnaires.

Un des points importants levés par Davis est que les algorithmes génétiques sont des méthodes d'optimisation d'une grande souplesse et très robustes, en particulier vis-à-vis du bruit, mais ne constituent quasiment jamais la meilleure méthode d'optimisation d'un problème donné.

Jusqu'ici, ce chapitre a introduit les algorithmes évolutionnaires, en général, et les algorithmes génétiques en particulier. Après une introduction d'abord pratique puis théorique à ces algorithmes, les points suivants devraient être retenus :

- le comportement d'un algorithme génétique est essentiellement stochastique. Comme toute méthode évolutionnaire, un algorithme génétique n'offre aucune garantie d'obtention de l'optimum global du problème considéré en un temps fini ;
- la conception d'un algorithme génétique pour un problème donné passe par la mise au point de nombreux opérateurs et paramètres. Une approche incorporant le plus de connaissances *a priori* du problème possible est souhaitable ;
- la grande liberté offerte à travers la conception des différents opérateurs accorde une grande souplesse aux algorithmes génétiques. Ces algorithmes permettent de traiter un très grand nombre de problèmes différents et ce, en accordant leurs différentes composantes à ces problèmes.

Nous nous sommes jusqu'alors intéressés aux fondements des algorithmes génétiques, la forme canonique de celui-ci, les opérateurs traditionnellement employés ainsi que les grands traits des théories associées. Mais depuis les deux dernières décennies, l'intérêt porté aux méthodes évolutionnaires a permis l'émergence de plusieurs méthodologies et approches liées à celles-ci. La suite de ce chapitre s'attache à en décrire certaines des plus reconnues.

5.4 Développements autour des algorithmes génétiques

Cette section n'a pas pour ambition d'étudier de manière exhaustive les nombreux concepts et méthodologies conçus autour des algorithmes génétiques. Nous souhaitons, ici, présenter certains des développements figurant parmi les plus connus et ayant trait à nos travaux.

Différentes aspirations furent à l'origine de ces développements :
– du désir de s'affranchir de la phase, parfois hasardeuse, du paramétrage de l'algorithme ;
– d'une adaptation de l'algorithme génétique à certains types de problèmes ;
– d'une extension du principe de parallélisme implicite de l'algorithme génétique ;
Le lecteur trouvera dans la suite de ce chapitre les différents concepts et applications de ces développements qui ont, pour certains, servis de base à notre travail.

5.4.1 Adaptativité des paramètres

Une utilisation efficace des méthodes évolutionnaires nécessite la détermination des paramètres tels que la taille de la population ou les différentes probabilités employées pour l'exploration ou l'exploitation de l'espace des solutions.
Les premiers algorithmes évolutionnaires étaient paramétrés de façon statique : les différents paramètres influant sur l'évolution de la population étaient déterminés par l'utilisateur, préalablement au lancement de l'algorithme et demeuraient fixes par la suite. Le réglage (comprendre par là le choix du jeu optimal de paramètres) s'effectuait de manière empirique, en choisissant les paramètres retournant les meilleurs résultats.

Le problème d'un tel réglage est qu'il revient généralement à affiner les choix des paramètres successifs en ignorant les interactions existant entre les opérateurs d'exploration (mutation) et de recombinaison (croisement) de l'algorithme. À l'opposé, tester sans stratégie des ensembles de paramètres revient à effectuer un nombre prohibitif d'essais.

Une limite évidente de cette technique est qu'un réglage statique, non content de nécessiter un nombre important d'essais, ne débouche pas forcément – bien au contraire – sur un choix globalement optimal des paramètres.

Enfin, l'utilisation de paramètres fixes au sein d'un algorithme à vocation évolutionnaire – donc dynamique – est plus que contradictoire, sans compter le fait qu'un choix de paramètre optimal lors d'une phase de l'évolution ne l'est plus forcément durant une autre.

L'exemple le plus intuitif illustrant l'utilité de paramètres dynamiques est celui de la probabilité de mutation. Au départ de l'exploration, une probabilité importante est souhaitable afin d'étendre celle-ci mais, à mesure que la population converge, une probabilité réduite est préférable afin d'affiner la recherche.

D'après [Eiben et al., 1999, Richter et Paxton, 2005], nous pouvons regrouper les différentes politiques d'adaptativité d'un algorithme génétique au sein d'une classification.

Contrôle déterministe : la valeur du paramètre est influencée selon une règle déterminée. Cela peut être une évolution en fonction du nombre d'itérations déjà effectuées, par exemple ;

Contrôle adaptatif : le paramètre est ajusté en fonction de l'évolution de la recherche : on utilise un retour sur la qualité de la population ou sur l'exploration en cours ;

Contrôle auto-adaptatif : les paramètres ne sont plus globaux mais locaux à chaque individu *i.e.* les paramètres font partie du génotype. La distinction avec le contrôle adaptatif réside dans le fait que les paramètres sont gérés comme le reste du génotype et donc soumis aux différents opérateurs génétiques.

Une des premières approches de l'adaptativité appliquée aux algorithmes évolutionnaires est due à Rechenberg avec la règle des 1/5 [Rechenberg, 1970]. Cette règle s'applique à l'origine dans le cadre d'une stratégie évolutionnaire notée (1+1) (un seul parent et un seul enfant sont évolués) :

- les solutions sont modélisées par un vecteur de nombres réels,
- chaque génération comporte un seul parent et un seul enfant,
- la mutation consiste en l'addition du vecteur solution original avec un vecteur de nombres aléatoires générés par une distribution gaussienne de moyenne 0 et d'écart type σ : $\mathcal{N}(0, \sigma)$.

La règle des 1/5 consiste à adapter le pas de la mutation, σ, en fonction des performances observées de la descendance par rapport à celle des parents. Précisément, on relève périodiquement (en fonction du nombre de paramètres évolués) le taux ϕ des mutations ayant généré un meilleur individu. S'il s'avère supérieur à 1/5, le pas de la mutation est augmenté ; inversement, le pas est diminué sinon. Le mécanisme de cette règle est résumé par l'équation 5.2.

$$\begin{array}{l} \phi > 1/5 \rightarrow \sigma = c \times \sigma \\ \phi = 1/5 \rightarrow \sigma = \sigma \qquad\qquad c > 1, d < 1 \\ \phi < 1/5 \rightarrow \sigma = d \times \sigma \end{array} \tag{5.2}$$

Le principe étant que, lorsque le taux ϕ est assez grand ($> 1/5$), cela signifie que l'on doit chercher à amplifier une recherche de toute évidence fructueuse. Inversement, si la recherche dégrade les solutions actuelles, il vaut mieux réduire le pas afin de chercher dans la proche périphérie de ces solutions. Cette stratégie a malheureusement été conçue sur la base de problèmes linéaires et ce, à travers une des premières formes de stratégies évolutionnaires consistant à ne faire évoluer qu'un seul parent pour un enfant à chaque génération. Des expériences ultérieures [Chellapilla et Fogel, 1999] montrèrent que la règle des 1/5 s'avérait rapidement inefficace et suscitait une convergence prématurée.

Bäck et Schultz [Bäck et Schütz, 1996] ont proposé de contrôler la probabilité de mutation en fonction du nombre d'itérations déjà effectuées. L'inconvénient d'une telle approche est de tenir compte uniquement du temps et jamais de l'évolution de la population (*i.e.* où en est cette dernière dans sa convergence ?).

Un autre exemple d'adaptativité de la probabilité de mutation de chacun des n différents bits encodant un individu est donné dans [Droste et al., 2001] :

$$\left\{ \begin{array}{l} P_{mute}(t) = 2 \times P_{mute}(t-1) \\ \text{si } P_{mute}(t) > \frac{1}{2}, \text{ alors } P_{mute}(t) \leftarrow \frac{1}{n} \end{array} \right\}$$

La probabilité de mutation d'un bit varie de façon cyclique, doublant à chaque itération de l'algorithme tout en restant dans l'intervalle $[\frac{1}{n}, \frac{1}{2}]$. La probabilité prend alors $\lfloor \log_2 n \rfloor$ valeurs différentes.

5.4.1.1 Contrôle par gain constant

Une possibilité proposée dans [Thierens, 2002] est le contrôle par gain constant. Le principe repose dans un apprentissage de forme stochastique de la valeur du paramètre.

Soit :
- I, un individu de la population, doté d'une probabilité de mutation P_{mute},
- ω, nommé *facteur d'exploration* : un paramètre réel fixé, supérieur à 1,
- κ, le *coefficient d'apprentissage*, un paramètre réel fixé, $1 < \kappa < \omega$.

Lors de sa phase de mutation, l'individu I est muté à trois reprises, suivant trois probabilités différentes : P_{mute}, P_{mute}/ω et $P_{mute} \times \omega$. Les trois nouvelles solutions obtenues sont évaluées et la meilleure d'entre elles rejoint la population.

La probabilité de mutation de l'individu ainsi sélectionné est fixée en fonction du facteur d'exploration à son origine et du coefficient d'apprentissage conséquent :

1. – $mutation(I, \frac{P_{mute}}{\omega}) \rightarrow (I_1, \frac{P_{mute}}{\kappa})$
   – $mutation(I, P_{mute}) \rightarrow (I_2, P_{mute})$
   – $mutation(I, \omega \times P_{mute}) \rightarrow (I_3, \kappa \times P_{mute})$
2. Choisir le meilleur individu de $\{(I, P_{mute}), (I_1, \frac{P_{mute}}{\kappa}), (I_2, \kappa P_{mute}), (I_3, \kappa P_{mute})\}$

Les valeurs préconisées pour les paramètres sont de $\omega = 1,5$ et $\kappa = 1,1$. L'inconvénient majeur de cette méthode est de nécessiter trois mutations et donc trois processus de génération/évaluation pour les trois différentes valeurs du taux ω.

5.4.1.2 Contrôle auto-adaptatif

Comme nous l'avons expliqué précédemment, le principe ici revient à encoder directement les paramètres de contrôle des opérateurs génétiques dans le génotype des individus. Il n'y a pas ici de lien direct entre le choix des paramètres et la qualité des individus retournés. Le lien se fait indirectement en se basant sur l'évolution de la population : les paramètres adéquats renvoient les meilleurs génotypes qui, à leur tour, se propagent dans la population.

Répartition normale de la probabilité de mutation

Un exemple, tiré de [Bäck et Schütz, 1996] est :

$$P_{mute}(t+1) = \left(1 + \frac{1 - P_{mute}(t)}{P_{mute}(t)} e^{-\gamma \mathcal{N}(0,1)}\right)^{-1} \tag{5.3}$$

Tel que $P_{mute}(t+1)$ suit une distribution de probabilités de densité :

$$f_{P_{mute}(t+1)}(x) = \frac{1}{\sqrt{2\pi}\gamma x(1-x)} e^{\frac{-\left(ln\frac{x}{1-x}-\chi\right)}{2\gamma^2}} \tag{5.4}$$

où $\chi = ln\frac{p}{1-p}$ et γ est un coefficient d'apprentissage servant à contrôler la vitesse d'adaptation ($\gamma = 0.22$ dans [Bäck et Schütz, 1996]).

Autres approches adaptatives

Il existe de nombreuses méthodes d'adaptation ou d'auto-adaptation ; le lecteur intéressé par ce sujet pourra consulter les articles et ouvrages suivants : [Anastasoff, 1999, Angeline, 1995, Bäck, 1992, Bäck et al., 2000, Bäck et Schütz, 1996, Gomez, 2004, Saravanan et al., 1995].

5.4.2 Algorithmes à estimation de densités

Certaines méthodes évolutionnaires se démarquent des approches vues jusqu'ici en faisant évoluer non plus une population, sous la forme d'un ensemble de solutions candidates, mais la distribution de probabilités des meilleures solutions de l'espace de recherche. Ces algorithmes, regroupés sous la dénomination d'EDA (pour *Estimation of Distribution Algorithms*) ont fait leur apparition dans le courant des années quatre-vingt dix [Mühlenbein et PaaB, 1996]. Notons cependant que antérieurement à ces travaux, des approches similaires avaient fait leur apparition [Zhigljavsky, 1991, Baluja, 1994].

Depuis, différentes stratégies ont été développées dans le domaine de l'estimation de distributions de probabilités, s'adaptant à des espaces continus ou employant des modèles graphiques probabilistes pour la modélisation des distributions.

Deux ouvrages [Larrañaga et Lozano, 2001, Lozano et al., 2006] offrent un panorama étendu des méthodes à estimation de distribution. Nous limitons ici notre étude à l'introduction des deux approches EDA les plus connues : UMDA *pour Univariate Marginal Distribution Algorithm*) et PBIL (*Population-Based Incremental Learning*), avant de préciser certaines contributions au domaine pour enfin aborder les applications de ces concepts à l'apprentissage de la structure d'un réseau bayésien.

Principe général

Une difficulté inhérente aux algorithmes génétiques est la détermination de nombreux paramètres, détermination revenant à un problème d'optimisation des dits paramètres.

Le principe d'un EDA est de faire évoluer non plus une population de solutions mais un vecteur de probabilités formulant la probabilité jointe des meilleures solutions.

Pour cela, les opérateurs de croisement et de mutation usuellement employés dans les algorithmes génétiques sont ici remplacés par une estimation de la distribution jointe des solutions prometteuses. Cette estimation est itérativement employée afin de générer de nouvelles solutions qui, elles-mêmes, servent à réestimer la distribution de probabilités.

Nb_p individus, générés aléatoirement, constituent une population de départ Pop_0. Dans un premier temps, Nb_{SE} individus sont sélectionnés parmi P_0 avec $Nb_{SE} \leq Nb_p$ afin de constituer un échantillon Pop_{l-1}^{SE}. Ces derniers individus, reflétant les meilleures solutions générées, vont servir à la mise à jour de la distribution de probabilités évoluée au sein de l'algorithme dans la phase suivante. Enfin, un nouvel échantillon P_1 est généré à partir de la distribution mise à jour. L'algorithme boucle ensuite entre mise à jour de la distribution et phase d'échantillonnage/sélection sur cette distribution jusqu'à un certain critère d'arrêt.

Les EDA peuvent être schématisés par le pseudo-code de l'algorithme 8.

Algorithme 8 Algorithme EDA général

$Pop_0 \leftarrow Nb_p$ individus générés aléatoirement
$l \leftarrow 1$
Répéter
  $Pop_{l-1}^{SE} \leftarrow Nb_{SE} \leq Nb_p$ individus sélectionnés depuis Pop_{l-1}
  $p_l(x) = p(x|Pop_{l-1}^{SE}) \leftarrow$ Estimation de la distribution de probabilité des individus
  $Pop_l \leftarrow$ échantillonner Nb_p nouveaux individus (la nouvelle population) depuis $p_l(x)$
  $l \leftarrow l + 1$
Tant que la condition d'arrêt n'est pas satisfaite

Cette description, volontairement générale, occulte plusieurs facteurs, tels que la méthode de génération de l'échantillon initial P_0 et la méthode de sélection des Nb_{SE} individus au sein de cet échantillon. La problématique majeure dans l'élaboration d'une telle méthode demeure cependant la procédure d'estimation de la distribution de probabilités des meilleures solutions.

Nous présentons par la suite deux méthodes à base d'EDA parmi les plus connues : les approches UMDA (pour *Univariate Marginal Distribution Algorithm*) et PBIL (pour *Population-Based Incremental Learning*). Remarquons que ces méthodes assument le fait que la distribution de probabilités jointe évoluée peut se factoriser simplement, ce qui n'est pas toujours le cas.

Algorithme UMDA

Cet algorithme, issu de [Mühlenbein, 1998] et décrit par l'algorithme 9, modélise la distribution de probabilités jointe en la réduisant au produit des distributions marginales indépendantes : $p_l(x) = \prod_{i=1}^{n} p_l(x_i)$ sur les différents $x_i, i \in 1 \ldots n$ composant la solution x.

Chaque distribution marginale est elle-même estimée à partir des fréquences marginales tirées de l'échantillon Pop_l^{SE} : la notation $\delta_j(X_i = x_i|Pop_{l-1}^{SE})$ employée dans l'algorithme 9 a pour valeur 1 lorsque la i^e composante du j^e élément de Pop_{l-1}^{SE} vaut x_i et 0, sinon.

Algorithme PBIL

Issu des travaux de [Baluja, 1994], la population d'individus est ici représentée par un vecteur de probabilités. Les individus sélectionnés dans l'échantillon sont employés afin de mettre à jour la distribution de probabilités dont ils sont issus. L'algorithme PBIL, dans sa forme originelle, cherche à optimiser un vecteur binaire à n dimensions. La population est par conséquent

Algorithme 9 Algorithme UMDA général

$l \leftarrow 1$
$Pop_0 \leftarrow Nb_p$ individus générés aléatoirement
Répéter
$Pop^{SE}_{l-1} \leftarrow Nb_{SE} \leq Nb_p$ individus sélectionnés depuis Pop_{l-1}

$$p_l(x) = p(x|Pop^{SE}_{l-1}) = \prod_{i=1}^{n} p_l(x_i) \leftarrow \prod_{i=1}^{n} \frac{\sum_{j=1}^{Nb_p} \delta_j(X_i = x_i|Pop^{SE}_{l-1})}{Nb_{SE}}$$

$Pop_l \leftarrow$ échantillonner Nb_p individus depuis $p_l(x)$
$l \leftarrow l + 1$
Tant que la condition d'arrêt n'est pas satisfaite

représentée par un vecteur de probabilités : $p_l(x) = (p_l(x_1), p_l(x_2), \ldots, p_l(x_n))$ où $p_l(x)$ est la distribution de probabilités jointe sur l'espace des solutions, à la l^e génération et $p_l(x_i), i \in 1, \ldots, n$ est la probabilité d'obtenir un 1 en la i^e composante d'une solution.

Le fonctionnement de PBIL, décrit par l'algorithme 10, est similaire à celui de l'algorithme UMDA excepté pour la phase d'évaluation de la distribution évoluée. La règle d'évolution s'effectue selon un paramètre α aussi appelé *taux de relaxation* :

$$p_{l+1}(x) = (1 - \alpha)p_l(x) + \alpha\left(\frac{1}{N_{SE}}\right)\sum_{k=1}^{N_{SE}} x^l_{k:N_{SE}} \quad , \alpha \in [0, 1]$$

On voit que la valeur de α détermine l'importance donnée dans le cadre de la réestimation de la distribution à l'échantillon considéré. Il convient de remarquer que l'approche PBIL, représentée dans l'algorithme 10, coïncide avec UMDA lorsque le paramètre α est égal à 1.

Algorithme 10 Algorithme PBIL général

Entrée: Un paramètre α : le degré de l'apprentissage

Initialiser un vecteur de probabilités $p_0(x)$
$l \leftarrow 1$
Répéter
échantillonner $x^l_1, x^l_2, \ldots x^l_{Nb_p}$ à partir de $p_{l-1}(x)$
évaluer et classer $x^l_1, x^l_2, \ldots x^l_{Nb_p}$
sélectionner les Nb_{SE} ($Nb_{SE} \leq Nb_p$) meilleurs individus $x^l_{1:Nb_p}, x^l_{2:Nb_p}, \ldots x^l_{Nb_{SE}:Nb_p}$
Mettre à jour les coefficients de $p_l(x)$:

$$p_{l+1}(x_i) = (1 - \alpha)p_l(x_i) + \alpha\frac{1}{Nb_{SE}}\sum_{k=1}^{Nb_{SE}} x^l_{k:Nb_p}$$

$l \leftarrow l + 1$
Tant que la condition d'arrêt n'est pas satisfaite

Autres modélisations de la distribution

Dans le cas des algorithmes UMDA et PBIL, la distribution jointe recherchée est définie sur un ensemble de variables indépendantes. Lorsqu'il existe des dépendances entre ces différentes variables, il est alors nécessaire de faire appel à une modélisation plus riche. Le cas de dépendances bivariables est ainsi traité par l'algorithme MIMIC (pour *Mutual Information Maximization for Input Clustering*) [Bonet et al., 1996], lequel modélise la distribution jointe à l'aide d'une chaîne reliant les différentes variables. Cette chaîne est évoluée, à chaque génération, en optant pour la permutation sur l'ordre défini par la chaîne permettant de minimiser la divergence de Kullback-Leibler entre la distribution définie par la chaîne permutée et la distribution représentée par les individus sélectionnés dans l'échantillon en cours.

Cependant, lorsque les dépendances entre les variables modélisées relèvent d'un ordre supérieur à deux, une chaîne, telle que celle définie par MIMIC, ne suffit plus. La modélisation peut alors s'effectuer par l'intermédiaire d'un réseau bayésien.

Ainsi, l'algorithme EBNA (pour *Estimation of Bayesian Network Algorithm* [Larrañaga et al., 2000] représente la distribution jointe $p_l(x)$ à travers un réseau bayésien appris (structure et paramètres) à partir d'une base de données (constituée des individus sélectionnés au sein de la population échantillonnée). L'apprentissage de la structure du réseau bayésien modélisant la distribution jointe donne alors lieu à autant de variantes qu'il existe d'approches pour l'apprentissage (par algorithme PC, ou par l'algorithme K2, par exemple). Dans la même optique, [Pelikan et al., 1999] proposent l'algorithme BOA (*Bayesian Optimization Algorithm*) qui recherche une bonne structure à l'aide d'un algorithme glouton employant le score BDE et ce, en restreignant la recherche aux structures présentant un nombre limité de parents par variable.

Les méthodes EDA ont connu de nombreuses évolutions et adaptations à des cas précis. Par exemple les algorithmes UMDA et PBIL se sont ainsi vu adapter à des espaces de recherche continus (respectivement $UMDA_c$ [Larrañaga et al., 2001] et $PBIL_c$ [Sebag et Ducoulombier, 1998]).

Nous invitons le lecteur à se reporter à [Larrañaga et Lozano, 2001] pour un panorama des adaptations d'algorithmes EDA aux cas continus et/ou présentant des dépendances complexes entre les composantes des solutions.

5.4.3 Techniques de *niching*

Un des principaux aspects des algorithmes évolutionnaires est de favoriser l'émergence et la survie du meilleur individu. Or, il est difficile de maintenir un équilibre entre le maintien d'une diversité génétique au sein de la population et ce favoritisme envers les individus les plus performants.

Un autre aspect des algorithmes génétiques est de permettre une prise en charge des problèmes dits *multimodaux*. Un problème multimodal est caractérisé par une fonction d'évaluation de ses solutions présentant plusieurs optima locaux et un ou plusieurs optima globaux.

Les méthodes dites de *niching* sont une approche particulière, dédiée à la résolution des deux aspects précédemment évoqués. Ces méthodes sont basées sur un principe initial visant à recréer le phénomène de niches biologiques.

Ce type de méthodes est particulièrement apprécié dans le cadre de l'optimisation multimodale [2], bien qu'elles ne soient pas réservées à cet usage exclusif et peuvent tout aussi bien être employées pour la résolution de problèmes unimodaux mais reconnus comme difficiles.

Un panorama complet de ces méthodes, de leurs caractéristiques et de leurs performances sur certains problèmes peut être trouvé dans [Mahfoud, 1995], par conséquent nous ne présenterons ici que certaines des stratégies les plus connues dans le domaine.

La détermination de niches requiert la définition d'un voisinage et, par conséquent, d'une distance sur un espace. Selon l'algorithme, cet espace peut être l'espace génotypique ou l'espace phénotypique. Dans le cas de l'espace génotypique, si le codage des individus a lieu sur l'espace binaire une distance employable est la distance de Hamming entre les deux représentations. Si l'espace employé est l'espace phénotypique, on définit alors une distance *ad hoc*, en fonction du problème considéré.

Les techniques de *niching* peuvent être divisées en deux grandes catégories : les techniques dites spatiales et les techniques temporelles.

Les techniques spatiales

Ici, la subdivision de la population s'effectue au sein de l'espace des solutions. Un optimum de la fonction d'évaluation définit une niche. Selon les approches (point abordé plus loin), le partage et l'occupation d'une telle niche s'effectue de plusieurs manières, l'essentiel étant d'éviter que l'ensemble des individus gérés par l'algorithme ne se retrouve concentré dans le proche voisinage d'un unique optimum de la fonction d'évaluation.

Les méthodes procédant à une répartition spatiale des individus se retrouvent aussi fréquemment sous la dénomination de méthodes de méthodes de *niching parallèle*

Ces techniques sont elles-mêmes divisées en deux grandes familles :

***crowding* :** [De Jong, 1975] est le premier à présenter un algorithme recourant à ce principe ; les individus nouvellement générés remplacent ceux, parmi un échantillon de la population, leur étant les plus proches. Cette similarité est évaluée ici au niveau du génotype (il est cependant possible, dans d'autres méthodes, de définir la similarité sur le plan phénotypique). Une autre méthode de *crowding* est le *deterministic crowding* [Mahfoud, 1992, Mahfoud, 1994]. Ici, le remplacement est effectué par des tournois binaires opposant les parents à leurs enfants suivant leur ressemblance sur le plan phénotypique ; le vainqueur du tournoi passant à la génération suivante.

***sharing* :** le principe de niches est essentiellement celui du partage de ressources limitées par un groupement d'individus. Dans le cadre des algorithmes génétiques, la ressource d'une niche est, de manière intuitive, la *fitness*.

[2] l'optimisation multimodale réfère à des problèmes d'optimisation où la fonction d'évaluation des solutions présente plusieurs optima locaux et un ou plusieurs optima globaux.

Ceci jette les bases des méthodes dites de *sharing*. Une des méthodes les plus représentatives de ce principe a été proposée par [Goldberg et Richardson, 1987] et revient à considérer la *fitness* d'un individu donné comme étant directement proportionnelle au nombre d'autres individus situés dans cette niche (et donc en deçà d'une distance déterminée dans l'espace considéré). La *fitness* modifiée, $f'(x)$ d'un individu x est alors calculée en fonction de sa *fitness* de base $f(x)$ et d'une somme de termes $share_x(i)$ dépendant des n individus partageant la niche avec x :

$$f'(x) = \frac{f(x)}{\sum_{i=1}^{n} share_x(i)}$$

La fonction $share_x$ prend ses valeurs dans l'intervalle $[0, 1]$, en fonction de la distance entre les individus i et x.

Techniques temporelles

La justification d'une approche différente des approches dites spatiales provient du fait que ces dernières nécessitent implicitement un nombre important d'individus afin de pouvoir détecter et maintenir un nombre conséquent de niches.

La complexité découlant d'un tel nombre, couplée au fait que le maintien des niches nécessite de nombreux calculs de distances entre les différents éléments de la population totale ont incité les auteurs de [Beasley et al., 1993] à développer une approche différente, le *niching séquentiel*.

En présence d'un problème d'optimisation multimodale, l'objectif de cette technique est de déterminer séquentiellement les multiples optima de la fonction d'évaluation utilisée. Pour cela, l'algorithme génétique est itéré à plusieurs reprises en modifiant à chaque itération la fonction d'évaluation ; cette modification revient à déprécier les zones de l'espace des solutions correspondant aux optima précédemment localisés. Au terme de chaque itération de l'algorithme génétique, le meilleur individu jusqu'alors trouvé est considéré comme un optimum et définit une niche dans son voisinage. En ce point de l'espace des solutions, la *fitness* est dégradée de façon à décourager toute future exploitation du même génotype qui est, lui stocké. Après ceci, l'algorithme est de nouveau itéré en employant la *fitness* modifiée.

Après un nombre défini d'itérations (ou d'optima recensés), les solutions identifiées comme des optima à la fonction d'évaluation sont retournées.

5.4.3.1 Discussion

Cette introduction aux méthodes de *niching* a permis de dégager deux approches principales, les approches spatiales et les approches temporelles. Si, d'après [Mahfoud, 1995], le *niching* spatial est plus efficace que le *niching* temporel, les méthodes spatiales impliquent un certain nombre de contraintes. La maintenance de plusieurs niches, suffisamment distinctes pour justifier une telle approche, nécessite un nombre important d'individus [Hu et Goodman, 2004]. D'autre part, l'appel au calcul de distances est aussi récurrent. Par conséquent, recourir à une approche spatiale de type *crowding* ou *sharing* implique d'avoir à traiter d'un problème avec une

fitness et une fonction de distance inter-individus pouvant être rapidement calculées. Le principal reproche adressé à l'égard des méthodes séquentielles est le fait de modifier les paysage de la *fitness* et donc de poser d'éventuels barrages à des évolutions ultérieures en interdisant ou du moins en contraignant fortement le réemploi de briques élémentaires correspondant à ces zones dépréciées.

Il faut aussi rappeller que la modification de la valeur de la fonction *fitness* pour un individu fait aussi partie du fonctionnement des algorithmes de *sharing*.

5.4.4 Algorithmes génétiques parallèles

Les algorithmes génétiques se sont naturellement prêtés, de par la nature de leur fonctionnement aussi bien que par les évolutions technologiques, à leur parallélisation. La parallélisation d'un algorithme génétique revêt généralement deux formes :

Parallélisation des calculs : pour certains problèmes, le coût des calculs liés à l'évaluation des individus est assez conséquent pour que l'on envisage de répartir ces calculs sur différentes machines,

Parallélisation de populations : dans ce cas précis, les individus sont répartis en des sous-populations évoluant en parallèle. Des échanges d'individus ou d'informations peuvent avoir lieu entre les sous-populations ainsi constituées.

5.4.4.1 Répartition des calculs liés à l'évaluation

Dans ce schéma, on dispose d'une machine maître sur laquelle est gérée l'évolution de la population de la même manière qu'avec un algorithme génétique simple. On dispose de même d'une série de machines reliées à la première et qualifiées d'esclaves qui, elles, prennent en charge le calcul des *fitness* des individus évoluant sur la machine maître.

La machine maître répartit alors les calculs des différentes évaluations nécessaires à chaque génération en transmettant les caractéristiques des individus aux machines esclaves. L'évolution de la population sur la machine maître reprend dès réception de la totalité des évaluations depuis les machines esclaves. La première implémentation d'une telle parallélisation appliquée à une méthode évolutionnaire est due à [Grefenstette, 1981].

Un problème évident de ce type de schéma est le temps pris par les différentes communications entre maître et esclaves, problème d'autant plus important que le nombre de machines sur lesquelles sont répartis les calculs est grand. Cependant, le schéma demeure avantageux dès lors que le temps du calcul de la *fitness* des individus est important par rapport au temps mis pour la communication des données.

5.4.4.2 Répartition des individus en sous-populations

Dans ce type de stratégie parallèle, les calculs sont eux aussi traditionnellement répartis mais le principe fondamental est de permettre une répartition de l'ensemble des individus

évolués en sous-populations, elles mêmes évoluées sur des machines séparées. Outre l'avantage représenté, en temps de calculs, par une répartition des individus sur plusieurs processeurs, une telle méthode permet d'évoluer des populations plus ou moins indépendantes et donc d'espérer une meilleure couverture de l'espace de recherche.

Plusieurs types de répartitions peuvent avoir lieu et on distingue généralement les méthodes répartissant des groupes de populations, que l'on désigne par le terme de méthodes de parallélisme à grains grossiers (*coarse-grained*) et les méthodes associant, idéalement, un seul individu par processeur que l'on désigne par le terme de méthodes de parallélisme à grain fin (*fine-grained*).

Les premières méthodes regroupent diverses stratégies visant à répartir les individus en sous-populations isolées. Les individus d'une sous-population évoluent de la même manière que pour un algorithme génétique classique et des stratégies de migration sont mises en place afin de permettre le transfert d'individus d'une population à une autre. Ces méthodes nécessitent alors la définition de paramètres tels que la fréquence ou la probabilité de migration ou la mise en place d'une stratégie de sélection de candidats à la migration dans une population donnée. Un exemple connu est le modèle en îlots proposé par [Cohoon et al., 1987]. Ce modèle, expliqué plus en détail dans une section ultérieure, répartit les individus sur plusieurs îlots, ponctuellement reliés entre eux par une stratégie de migration. Le lecteur intéressé pourra trouver une bonne introduction à différentes stratégies dans [Cantu-Paz, 1997].

Un modèle similaire est celui du *stepping stone* (que l'on pourrait traduire, grossièrement, par *tremplin*) présenté par [Mühlenbein, 1991]. Ici, les processus migratoires sont limités aux localisations immédiatement voisines. De plus, un processus d'amélioration locale par l'intermédiaire d'une descente de gradient intervient sur les individus d'une population dès lors que la qualité de celle-ci stagne pendant un certain nombre d'itérations.

Les modèles à grain fin différent des méthodes précédentes, comme nous l'avons évoqué, de par le fait que les individus d'une population sont répartis en groupes de très petite taille (parfois, un seul individu) eux-mêmes répartis sur les différents processeurs. Ce type de méthode procède, à la manière des modèles en îlots, par des communications et interactions entre non plus des populations mais des individus ou petits groupes d'individus voisins. Ces modèles se retrouvent aussi sous la dénomination de modèles *cellulaires*. Un exemple de parallélisme à grain fin peut être trouvé dans [Spiessens et Manderick, 1991] où les auteurs mettent en place un système cellulaire où chaque cellule/individu est associée à un processeur.

Les méthodes de parallélisation, et plus particulièrement le modèle de populations réparties en îlots, seront abordées plus en détail dans la suite de ce travail.

5.5 Applications à l'apprentissage de structures

C'est avec les travaux de Larranaga [Larranaga et al., 1996, Etxeberria et al., 1997] qu'apparaissent les premières tentatives d'apprentissage de structure par un algorithme génétique. L'auteur tente alors d'effectuer l'apprentissage avec ou sans connaissance d'un ordre topologiquement correct sur les variables du réseau à l'aide d'un algorithme génétique.

Ce premier article présentait surtout l'intérêt d'ouvrir la voie pour des études ultérieures ; en effet, le choix des paramètres et des opérateurs était limité : taille de population limitée, opérateur de croisement en un point. De plus les possibilités et contraintes inhérentes à l'espace des équivalents de Markov étaient alors peu employées.

Dans [Larrañaga et al., 1996], les auteurs emploient un algorithme évolutionnaire de manière indirecte. En effet, celui-ci effectue sa recherche sur l'ensemble des ordres topologiques. Une fois le meilleur ordre déterminé, celui-ci sert de base à un apprentissage par l'algorithme K2 (cf. section 4.4.2). Les auteurs emploient des opérateurs couramment employés pour la résolution du problème du voyageur de commerce (PVC), en effectuant toutefois quelques modifications. En effet, si un chemin peut être symétrique, cela n'est pas le cas avec un ordre topologique (i.e. les chemins $A \rightarrow B \rightarrow C$ et $C \rightarrow A \rightarrow B$ sont une même solution pour le PVC tandis que les ordres topologiques $(1, 2, 3)$ et $(3, 1, 2)$ sont deux solutions au problème de l'apprentissage de structures).

Cotta et Muruzábal [Cotta et Muruzábal, 2002] soulignent l'intérêt de l'emploi d'opérateurs phénotypiques plutôt que génotypiques *i.e.* prenant en compte l'expression même, dans l'individu de l'allèle pris en compte plutôt que par une sélection purement aléatoire (approche génotypique).

Dans [Wong et al., 1999], le critère MDL est employé pour l'apprentissage des structures. Leur algorithme, nommé MDLEP (pour dénoter la combinaison du critère MDL à la programmation évolutionnaire) ne comporte pas d'opérateur de croisement mais utilise en revanche une série d'opérateurs de mutation afin de faire évoluer la population courante. A noter que l'un des opérateurs est "guidé" dans le choix des gènes à muter par l'apport de ceux-ci en terme de critère MDL, celui-ci étant calculé au départ pour chaque arc possible.

Par la suite, les auteurs développèrent une version avancée de MDLEP, nommé HEP (*Hybrid Evolutionary Programming*) pour laquelle une hybridation est mise en place [Wong et al., 2002]. Celle-ci consiste à déterminer préalablement un squelette en effectuant une série de tests d'indépendance d'ordre faible (0 et 1) permettant de limiter l'espace de recherche : si une variable X est indépendante d'une variable Y suite aux tests, les arcs $X \rightarrow Y$ et $X \leftarrow Y$ ne pourront être ajoutés par l'opérateur de mutation. À cela, les auteurs ajoutent, dans un souci de rapidité de calcul, la fermeture des opérateurs en interdisant la création, consécutivement à une mutation, d'un circuit au sein d'un individu. Les résultats se montrent dès lors substantiellement meilleurs qu'en l'absence d'hybridation, même si les comparaisons se limitent à une confrontation entre les deux versions de leur algorithme.

Une autre approche, similaire à la précédente, est celle adoptée dans [van Dijk et al., 2003a, van Dijk et al., 2003b, van Dijk et Thierens, 2004]. Les auteurs proposent une méthode basée elle aussi sur la construction préalable d'un squelette à partir de tests d'indépendance statistiques d'ordre 0 et 1. Les gènes sont alors les arcs non orientés du squelette et les allèles pour chaque gène sont au nombre de 3 : "absent", "$X \rightarrow Y$" et "$X \leftarrow Y$". Cette méthode est très semblable à celle de [Wong et al., 2002] excepté que l'évolution des individus ne se fait ici que par le biais des recombinaisons et de l'intervention de deux opérateurs de réparation. Les résultats de [van Dijk et Thierens, 2004] se montrent sensiblement meilleurs que ceux obtenus par l'algorithme HEP mais la qualification des résultats en termes de qualité structurelle (*i.e.* la structure obtenue est-elle proche de celle recherchée ?) n'est pas mentionnée.

Les auteurs de [Muruzábal et Cotta, 2004, Muruzábal et Cotta, 2007] ont tenté d'effectuer la

recherche directement sur l'espace des équivalents. La recherche s'effectue à la fois sur l'espace des graphes représentants et au sein de chacun de ces graphes par des orientations différentes dans l'espace des structures des arcs non-orientés des représentants. Le consensus à l'issue de cette recherche étant qu'une telle recherche duale ne s'avère efficace que lorsque le changement d'espace intervient au moment opportun.

Une autre approche de l'exploitation de l'espace des graphes essentiels est celle présentée dans [Acid et de Campos, 2003] où l'algorithme parcourt l'espace des graphes partiellement orientés sans circuit (*GPOSC*) *restreints*. Ces graphes sont une forme particulière de *GPOSC* dont plusieurs membres distincts peuvent correspondre à la même classe d'équivalence, occupant ainsi une position intermédiaire entre l'espace des *GOSC* et celui des *GE*.

Certains travaux appliquent les approches de type EDA, présentées précédemment, à la recherche de structures de réseaux bayésiens. Dans [Blanco et al., 2003], les auteurs se sont attachés à appliquer deux approches, de type UMDA et PBIL, à la recherche dans l'espace des *GOSC*. Ces algorithmes ont été appliqués à la distribution des arcs dans la matrice d'adjacence de la structure recherchée. Les résultats semblent en faveur de l'approche PBIL mais on peut cependant remarquer que certains résultats (comme par exemple le fait que les réseaux appris sans connaissance de l'ordre topologique soient parfois meilleurs que lorsque celui-ci est connu) auraient mérité une étude plus poussée ou du moins quelques commentaires.

Dans [Romero et al., 2004], deux approches, UMDA et MIMIC ont été appliquées à la recherche sur l'espace des ordres topologiques possibles pour la structure recherchée, les individus (c'est-à-dire les ordres topologiques candidats) étant eux-mêmes évalués par le biais du score bayésien. Enfin, un mot sur la recherche de structures en présence de données incomplètes. Le principal obstacle dans le cadre de cette problématique est que, comme dans le cas de l'algorithme Structural EM, l'évaluation d'une structure nécessite l'évaluation des paramètres associés à celle-ci. Si cette évaluation se fait par le biais de l'algorithme EM, le coût en calculs de l'évaluation est trop élevé pour pouvoir tirer profit d'un algorithme évolutionnaire. En revanche, une autre approche a été formulée par [Myers et al., 1999] proposant de ne faire non pas seulement évoluer les structures candidates mais aussi les données manquantes. L'évaluation des structures se faisant alors sur la base complétée.

5.6 Conclusion

Depuis leur apparition, les méthodes évolutionnaires ont connu de nombreux développements, tant théoriques que pratiques. De même, ces méthodes, souvent performantes pour des problèmes d'optimisation reconnus comme difficiles, se sont vues appliquées au problème de l'apprentissage de structures de réseaux bayésiens.

L'apprentissage de la structure d'un réseau bayésien à l'aide d'une fonction de score et en l'absence d'*a priori* tel que l'ordre topologique de la structure recherchée, il courant d'avoir recours à un algorithme de type glouton sur l'espace des structures ou bien sur celui des classes d'équivalences.

Le principal inconvénient de ces méthodes gloutonnes est de se retrouver fréquemment bloquées en une solution correspondant à un optimum local de la fonction d'évaluation. Ceci en raison de la présence de nombreux optima locaux dans l'espace des solutions. Optima d'autant plus nombreux que la base de cas servant à l'apprentissage est limitée en taille – les structures vraisemblables vis-à-vis de cette base étant alors nombreuses –.

La principale raison de cette convergence prématurée est qu'un algorithme glouton ne considère à tout moment qu'un unique point de l'espace des solutions. En l'absence, dans le voisinage de ce point, d'une solution mieux évaluée, la recherche s'arrête. La manière la plus répandue de contrevenir à cette situation est alors de recourir à plusieurs initialisations de l'algorithme glouton, à partir de structures initiales différentes les unes des autres, et de retenir la meilleure solution obtenue. Cette technique présentant l'inconvénient de décupler, bien sûr, les temps de calculs mais aussi de n'offrir aucune garantie quant à l'obtention de x solutions distinctes pour x initialisations différentes de l'algorithme glouton.

Les algorithmes évolutionnaires présentent deux avantages majeurs lors du traitement d'un problème présentant de nombreux optima locaux. D'une part, ils permettent de maintenir une population de solutions, i.e. plusieurs points de l'espace des solutions, idéalement diverses. Le maintien et l'évolution de différentes solutions et donc d'autant de points au voisinage desquels explorer l'espace des solutions, permet alors de réduire les chances de se retrouver bloqué en un unique point localement optimal. D'autre part, le comportement stochastique de ces méthodes, par le biais de l'opérateur de mutation, permet d'amplifier cette robustesse vis-à-vis de la présence d'optima locaux (sous condition de l'emploi de paramètres et d'opérateurs adaptés) en autorisant une exploration de l'espace des solutions qui n'est plus limitée au voisinage immédiat des individus de la population.

Nous avons choisi d'orienter notre travail vers le développement d'une méthode évolutionnaire adéquate afin de bénéficier non seulement des propriétés exploratoires d'une telle approche mais aussi de leur souplesse reconnue en adaptant les différents opérateurs à notre problème.

Le chapitre suivant décrit un algorithme génétique permettant de trouver une structure de bonne qualité à partir d'une base d'apprentissage de taille limitée. Certains problèmes inhérents à l'utilisation d'un algorithme évolutionnaire, tel que le nombre de calculs requis par l'évaluation des solutions, seront traités en exploitant les propriétés de décomposition de la fonction d'évaluation employée. Par la suite, nous avons enrichi cet algorithme en lui adjoignant différentes stratégies permettant une meilleure exploration de l'espace des solutions, par modification du paysage de la *fitness* employée, par une répartition spatiale des individus ou encore par un système d'adaptativité de l'opérateur de mutation.

Deuxième partie

Apprentissage de la structure d'un réseau bayésien par un algorithme évolutionnaire

Chapitre 6

Apprentissage avec répartition dans l'espace des solutions

Les algorithmes génétiques, appliqués à la recherche de structure de réseaux bayésiens, posent deux problèmes :

- la contrainte sur l'absence de circuits dans les structures crée un lien fort entre les différents gènes - et allèles - d'un individu, quelle que soit la représentation choisie. Les opérateurs, dans l'idéal, devraient tenir compte de ce fait ;
- il n'est pas rare qu'une heuristique de parcours de l'espace des solutions (algorithme génétique, méthode gloutonne, etc.) se retrouve bloquée sur un optimum local. Il est alors difficile de trouver un équilibre entre un parcours disparate pouvant s'affranchir de ce problème, au risque de négliger de nombreuses solutions de qualité, et un parcours plus minutieux ayant de grandes chances de ne retourner qu'une solution localement optimale.

Si le premier point implique essentiellement la conception d'une méthode évolutionnaire réfléchie et adaptée au problème, le deuxième point caractérise un problème relevant de l'optimisation multimodale. Pour ce type de problématique, nous avons vu qu'il existe une méthodologie particulière : le *niching* (cf. section 5.4.3). Avant d'étudier la mise en place d'une telle stratégie, nous allons procéder à une description préalable d'un algorithme génétique adapté à la recherche d'une bonne structure pour un réseau bayésien. Puis, nous nous attacherons à étudier et définir une stratégie de *niching* adéquate pour notre problème. Enfin, nous développerons une méthode aspirant à combiner les qualités des approches séquentielles à celles des approches spatiales.

6.1 Algorithme génétique simple

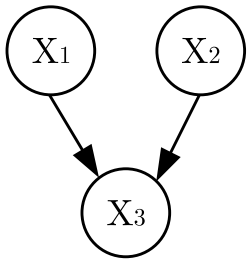
Les opérateurs, stratégies et paramètres présentés dans cette section sont les implémentations des éléments classiquement définis au sein d'un algorithme génétique. Avant de préciser le cœur de notre méthode de *niching*, nous détaillons ici la structure générale du moteur évolutionnaire. Par la suite, nous comparerons les résultats obtenus par cet algorithme avec ou sans application d'une stratégie de *niching*.

6.1.1 Définition d'un individu

L'algorithme génétique effectue une recherche dans l'espace des graphes orientés sans circuits. Chaque solution envisageable est représentée, dans la population, par sa matrice d'adjacence $C = (c_{ij}), \quad (i, j) \in [1 \dots N]^2$:

$$c_{ij} = \begin{cases} 1 & \text{si } X_j \text{ parent de } X_i \\ 0 & \text{sinon} \end{cases}$$

La figure 6.1 montre un exemple de correspondance entre une structure et sa matrice d'adjacence.



$$C = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

Figure 6.1 – Exemple de réseau bayésien et de la matrice d'adjacence correspondante.

6.1.2 Mesure de la qualité d'un individu

La qualité d'un individu est égale au score qu'il obtient vis-à-vis de la base de cas considérée. Nous avons vu, dans la partie consacrée à l'évaluation des méthodes présentées, que plusieurs critères étaient susceptibles d'être employés pour évaluer la qualité d'une structure. Dans la littérature, la grande majorité des méthodes emploient, dans leurs résultats, le score BDeu ou le critère BIC, mais ces deux mesures sont asymptotiquement égales. Nous avons finalement choisi d'employer le critère BIC et ce pour deux raisons :

- le critère BIC choisit le modèle le plus simple parmi ceux décrivant la distribution de probabilité recherchée. Si dans le cas des bases d'apprentissage d'effectifs faibles ceci peut entraîner la non-détection de certains liens, il nous semble plus logique d'admettre un ensemble d'omissions sur les dépendances du domaine que d'en ajouter de superflues ;
- la plupart des méthodes de la littérature emploient ce critère. Si les mesures ne sont pas comparables numériquement du fait que les différents travaux emploient généralement des bases d'apprentissage non publiques, les résultats qualitatifs sont en revanche plus facilement comparables.

De plus, il faut remarquer que le critère BIC ainsi que le score BDeu approximent la vraisemblance marginale de la structure et donc, à mesure que la taille de la base d'apprentissage augmente, ces deux scores convergent vers une même valeur.

6.1.3 Initialisation des individus

En nous basant sur les résultats de [François et Leray, 2004] et d'après nos propres expérimentations, nous avons opté pour une initialisation de la population de structures par les différents arbres (selon le sommet racine choisi) retournés par l'algorithme MWST (cf. section 4.4.1). Bien que ces n arbres soient Markov-équivalents, cette initialisation permet de générer, au point de vue de la population évoluée par l'algorithme génétique, des individus présentant des caractéristiques pertinentes (sous la forme d'un sommet prédécesseur pour chaque variable, excepté la racine de l'arbre). De plus, dès les premières générations, l'action conjuguée des opérateurs de croisement et de mutation, décrits plus loin, permet d'obtenir des individus variés et de bonne qualité et enfin d'obtenir un gain intéressant en terme de temps de convergence.

Nous utilisons l'arbre non-orienté retourné par l'algorithme : chaque individu de la population est initialisé par un arbre orienté à partir d'une racine choisie aléatoirement. Ce mécanisme permettant d'introduire une certaine diversité dans la population.

6.1.4 Stratégies et paramètres de sélection

Nos premiers essais ont montré que l'emploi d'une sélection par roulette – où la probabilité qu'a un individu d'être sélectionné pour la reproduction est directement proportionnelle à sa qualité – débouchait sur une convergence prématurée de l'algorithme. Cette observation est assez commune. La sélection par roulette est effectivement le premier opérateur de sélection à avoir été proposé en algorithmique évolutionnaire et son principal défaut est d'être rapidement biaisé en accordant une trop grande importance aux individus les plus performants de la population.

Une autre possibilité est la sélection par tournoi. Cet opérateur maintient une pression constante, uniquement basée sur les rangs respectifs des individus considérés. Néanmoins, la pression reste forte, même dans le cas d'un tournoi binaire (le plus simple juste derrière la sélection aléatoire).

Une alternative permettant d'atténuer ce phénomène de prédominance de certains individus est le *fitness scaling*. Ces méthodes ainsi que leur principe général sont présentées dans [Forrest, 1985, Kreinovich et al., 1993] et ont pour objectif de permettre dans un premier temps d'empêcher le phénomène de prédominance des "super individus" lors des premières générations tout en assurant, lorsque la population converge, que les individus de qualité moyenne – alors en grand nombre – n'entrave pas la reproduction des meilleurs. Le principe général du *fitness scaling* revient à employer non plus la *fitness* des individus mais une fonction de celle-ci (fonction linéaire, en exposant ou exponentielle, pour citer les fonctions les plus répandues).

L'inconvénient des méthodes de *fitness scaling* est d'être définies et employées de manière *ad hoc*, *i.e.* choisies et employées de manière empirique, sans pour autant s'appuyer sur une réelle étude ou justification théorique de leur efficacité.

Nous avons opté pour une sélection par rang, où chacun des λ individus de la population a une probabilité de se reproduire égale à :

$$P_{select} = 2 \times \frac{\lambda + 1 - rang(individu)}{\lambda \times (\lambda + 1)} \tag{6.1}$$

Cette stratégie permet de favoriser les individus les mieux adaptés tout en laissant aux individus les plus "faibles" l'opportunité de participer au processus d'évolution.
Si l'inconvénient majeur de cette méthode est de nécessiter un classement systématique des individus au préalable, ce coût reste négligeable.

Par la suite, les individus sélectionnés ont une probabilité P_{cross} de se reproduire. Au cours de nos tests, nous avons utilisé une valeur de P_{cross} égale à 0,8.
Enfin, l'opérateur de mutation est, quant à lui, appliqué aux individus issus de la phase de croisement ainsi qu'aux individus n'ayant pas été sélectionnés.
Lors du passage d'une population P_t de taille λ à la population P_{t+1} suivante, nous employons une stratégie de remplacement qualifiée *d'élitiste* ; une telle approche revient à conserver le meilleur individu de la population à l'instant t lors du passage à l'instant $t+1$, dès lors qu'aucun des individus nouvellement générés (par croisement et mutation) à l'instant t ne s'est avéré meilleur que celui-ci – le reste de la population au temps $t+1$ étant alors constitué des $\lambda - 1$ meilleurs nouveaux individus –.

6.1.5 Opérateurs génétiques

Plusieurs opérateurs interviennent dans l'action de l'algorithme génétique, dont les opérateurs de croisement et de mutation. Nous avons aussi dû développer un opérateur de réparation afin de pouvoir éliminer les circuits éventuellement créés lors de l'évolution de la population.

Opérateur de croisement

Les possibilités sont, ici aussi, nombreuses. Dans un premier temps, nous avions opté pour un opérateur de croisement simple, en un point [Delaplace et al., 2007a] :
Soient P1 et P2, deux individus choisis pour être mutuellement croisés ;

- l'opérateur coupe les matrices d'adjacence de P1 et P2 sur les colonnes, au point $\left\lfloor \frac{n}{2} \right\rfloor$;
- le premier enfant issu du croisement portera les colonnes $(1 : \left\lfloor \frac{n}{2} \right\rfloor)$ de $P1$ et $(\left\lfloor \frac{n}{2} \right\rfloor + 1 : n)$ de $P2$, comme indiqué sur la figure 6.2.

Par la suite, un autre opérateur a été développé, sur le modèle de [Vekaria et Clack, 1998]. Cet opérateur permet de générer lui aussi deux individus mais, à la différence du précédent, le choix des points de croisement est une fonction de la qualité de l'individu [Delaplace et al., 2007b].

La forme prise par le critère BIC (et, en général, par toute fonction de score décomposable) permet en effet d'attribuer un score local à chaque ensemble $\{X_i, \Pi_i\}$ du domaine $\mathcal{U}$. Nous pouvons donc choisir, à partir de ces différents scores locaux, de générer un individu ayant reçu les meilleurs éléments de ses ancêtres – ainsi que, respectivement, un individu constitué des éléments les moins performants –.

Le schéma de fonctionnement de cet opérateur, présenté dans la figure 6.3 est semblable à celui de l'opérateur simple à ceci près que les deux enfants générés reçoivent de chacun des graphes parents les ensembles sommets/variables parents ayant les meilleures évaluations locales – respectivement les plus mauvaises pour le deuxième graphe enfant –.

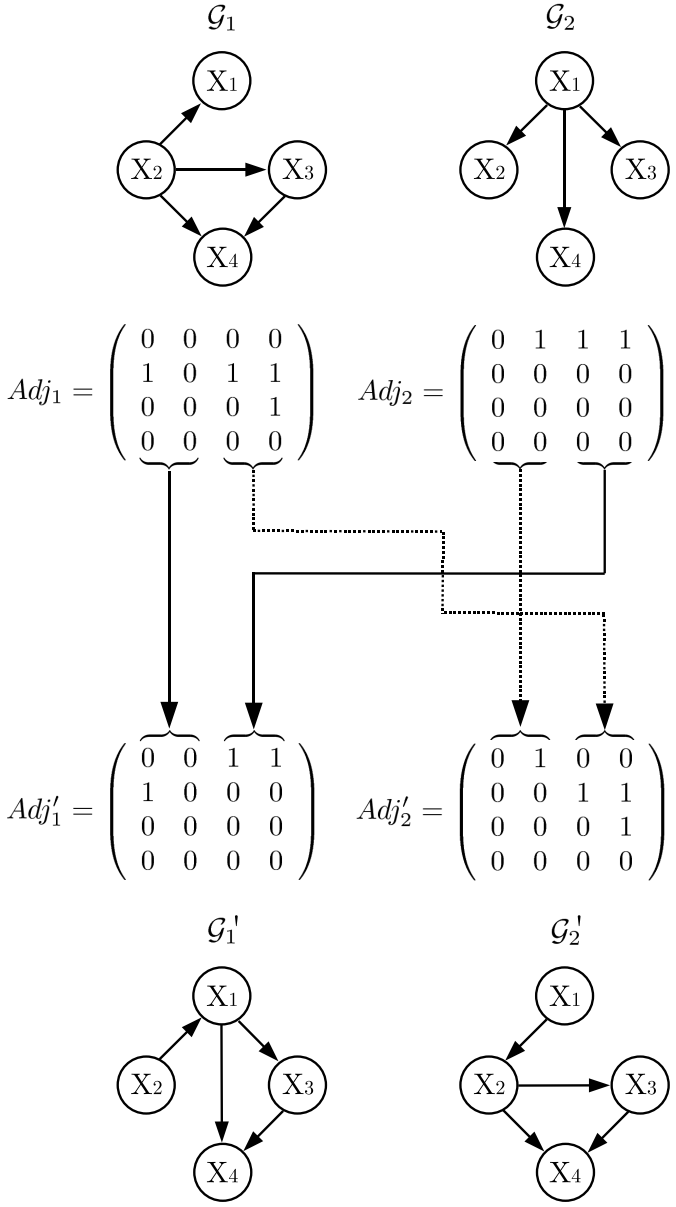


Figure 6.2 – Exemple de croisement en un point.

Dans l'exemple de la figure 6.3, nous admettons les inégalités suivantes :

$$\begin{array}{rcl} S_{Adj_1}(X_1) & \geq & S_{Adj_2}(X_1) \\ S_{Adj_1}(X_2) & \geq & S_{Adj_2}(X_2) \\ S_{Adj_1}(X_3) & \geq & S_{Adj_2}(X_3) \\ S_{Adj_1}(X_4) & \leq & S_{Adj_2}(X_4) \end{array}$$

où $S_{Adj_k}(X_i)$ représente le score obtenu localement en X_i sur le graphe $\mathcal{G}_k$. Nous obtenons alors deux individus : un composé des ensembles de variables parents les mieux évalués et l'autre composé des ensembles restants.

La composition du meilleur individu étant la suivante :

1. le "meilleur" descendant correspond en premier lieu à une copie du meilleur parent ;
2. dans l'ordre et depuis la première variable, chaque ensemble de sommets prédécesseurs Π de la variable X_i considérée est remplacée par l'ensemble correspondant, issu du plus mauvais parent, à condition que :
   – le score local obtenu en X_i pour le plus mauvais parent est strictement meilleur qu'au sein du meilleur parent ;
   – le remplacement ne génère pas de circuit au sein du descendant.

La procédure étant exactement l'opposée, en considération des scores locaux – devant alors être qualitativement inférieurs – pour la génération du plus mauvais descendant.

Nous constatons qu'une des conditions à l'échange des ensembles de sommets prédécesseurs Π_i est la non-génération de circuits au sein des descendants, suscitant ainsi une fermeture de l'opérateur.

Un exemple d'un cas de création d'un circuit est décrit par la figure 6.4 dans laquelle deux *GOSC* équivalents (et donc de même score) présentent comme seule différence un arc, symbolisé par une arête dans le graphe équivalent $\mathcal{G}_{eq}$ de ces deux *GOSC*. L'opérateur de croisement va alors générer deux *GOSC* descendants : un ne présentant aucun arc et un deuxième présentant un circuit localisé entre les deux variables. On peut souligner le fait que cette situation peut aussi se présenter dans le cas de deux graphes parents non équivalents mais présentant une configuration similaire à celui de l'exemple sur deux ensembles de variables parents.

La fermeture de l'opérateur permet ainsi de gagner en rapidité ainsi que d'éviter, comme dans l'exemple de la figure 6.4, de perdre une information (l'arête est présente chez les deux parents mais seulement chez un enfant).

Opérateur de mutation

Chaque colonne $j \in [1 \dots n]$ d'un individu a une probabilité P_{mute} d'être modifiée. Si une colonne mute, une modification parmi celles possibles est effectuée en un de ses coefficients c_{ij} :

- Si $c_{ij} = 0$, gain d'un parent : $c_{ij} \leftarrow 1$;
- Si $c_{ij} = 1$, deux possibilités, équiprobables :
  – perte d'un parent, $c_{ij} \leftarrow 0$;
  – ou inversion de la parenté ($c_{ij} \leftarrow 0, c_{ji} \leftarrow 1$).

Par modifications possibles, nous entendons bien sûr l'ajout d'arcs alors inexistants et, inversement, la soustraction ou l'inversion d'arcs existants ; mais aussi le fait que l'on ne peut ajouter un arc arrivant sur une variable depuis une variable enfant de celle-ci. Cela créerait un circuit de longueur 2 et, après appel à l'opérateur de réparation, reviendrait à une inversion tout en ayant nécessité plus de calculs.

Phase d'évaluation des individus

L'algorithme génétique tire parti de la décomposition de la fonction d'évaluation et évalue les nouveaux individus dès leur création, par croisement, mutation ou réparation. L'impact de

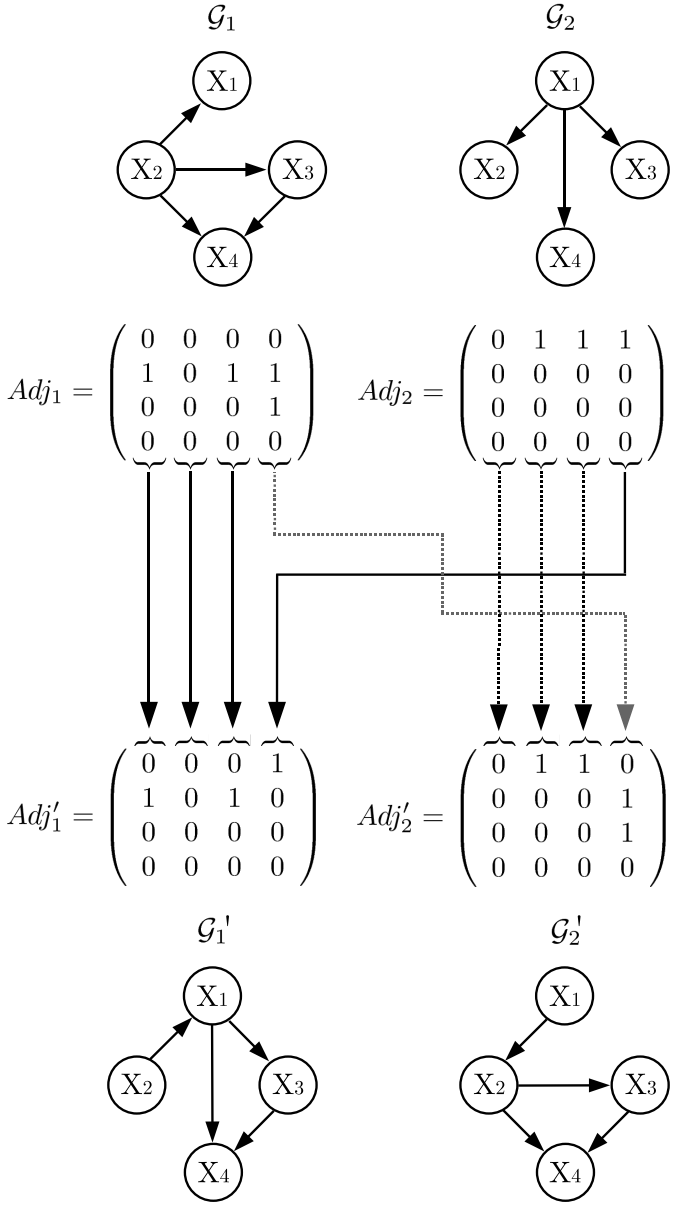


Figure 6.3 – Exemple de croisement sélectif en plusieurs points. Ici, les scores locaux du graphe parent Adj_1 sont supérieurs à ceux au sein du graphe parent Adj_2, excepté pour la variable X_4.

toute modification locale sur le génome d'un individu est immédiatement répercutée sur le phénotype de celui-ci par le biais du calcul du score local. La conséquence direct de ceci est que la phase d'évaluation de la population générée, présente dans l'algorithme génétique canonique vu dans la section 5.2, a en fait lieu pour chaque individu, en fonction des modifications opérées, à la suite des mutations subies par celui-ci.

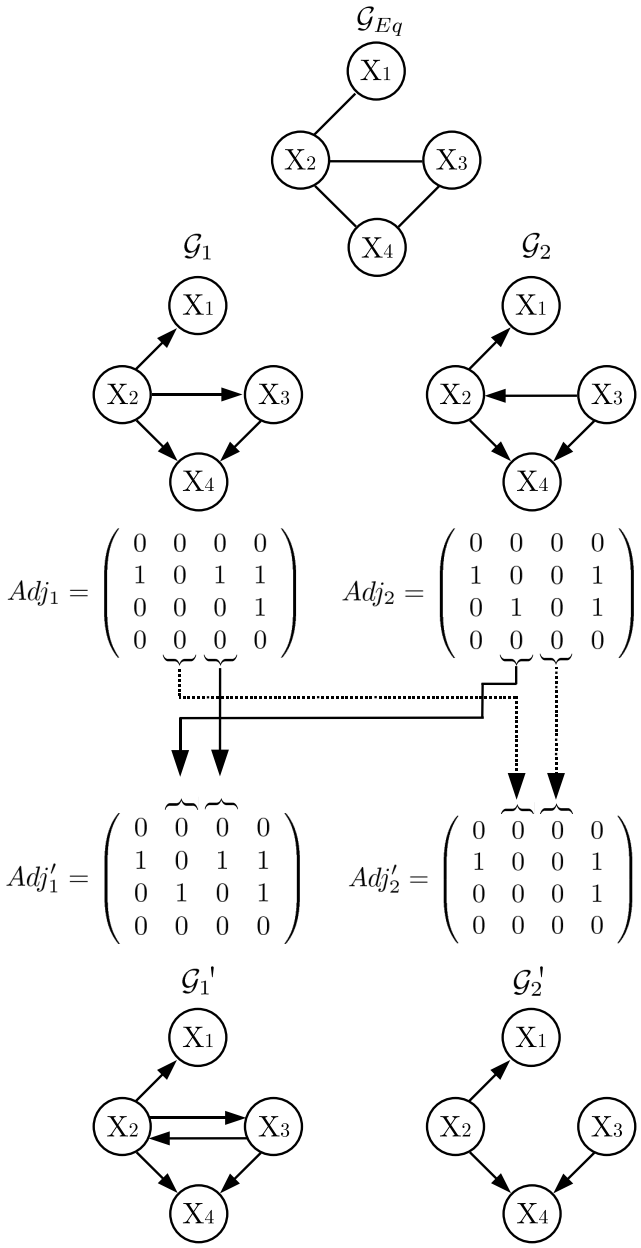


Figure 6.4 – Exemple de création de circuits par l'opérateur de croisement sélectif.

Opérateur de réparation

Si, à la suite du croisement ou de la mutation, un graphe orienté présente un circuit, cet individu n'est pas viable. Dans ce type de situation, deux choix sont couramment offerts : éliminer l'individu concerné ou bien tenter de le réparer. Dans notre algorithme, un détecteur de circuits est appliqué aux individus mutés. Si un circuit est détecté, l'opérateur de réparation supprime un des arcs appartenant au circuit.

Nous avons choisi d'indiquer à l'opérateur quels étaient les arcs dont la suppression était la

plus judicieuse.
Dès l'initialisation, l'algorithme calcule l'information mutuelle, telle qu'elle est définie dans [Chow et Liu, 1968], entre chaque paire de variable (X_1, X_2), du domaine :

$$W(X_1, X_2) = \sum_{x_1,x_2} \frac{N_{x_1,x_2}}{M} \log \frac{N_{x_1,x_2} \times M}{N_{x_1} \times N_{x_2}} \tag{6.2}$$

où l'on note :
- N_{x_1,x_2}, le nombre d'occurences simultanées dans la base de $X_1 = x_1$ et $X_2 = x_2$;
- N_{x_1}, le nombre d'occurences dans D de $X_1 = x_1$;
- N_{x_2}, le nombre d'occurences dans D de $X_2 = x_2$.

L'information mutuelle nous fournit une indication quant au degré de dépendance des deux variables X_A et X_B. Notamment, cette information est nulle si et seulement si les deux variables sont indépendantes.

A chaque appel, l'opérateur de réparation récupère la valeur de l'information mutuelle des couples de variables du circuit détecté ; l'arc effectivement supprimé étant alors celui reliant le couple de variables détenant l'information mutuelle la plus faible.

Il peut arriver qu'un individu présente plusieurs circuits, ceci à la suite d'une mutation ayant généré et/ou inversé plusieurs arcs. Dans ce cas, la réparation est effectuée de manière itérative, en commençant par supprimer le circuit le plus court jusqu'à ce que l'ensemble des circuits ait été supprimé.

Nous nous sommes jusqu'ici attachés à décrire les spécificités d'un algorithme génétique que l'on pourrait qualifier de *simple*. Hormis les opérateurs et certaines caractéristiques, cet algorithme n'applique aucune stratégie de parcours de l'espace de recherche ou de répartition de la population évoluée. La section suivante nous amène à réfléchir aux caractéristiques d'une stratégie de type *niching* qui serait adaptée au problème d'apprentissage de structures.

6.2 Choix d'une stratégie adaptée

Nous avons vu, dans la section 5.4.3, que les méthodes de *niching* étaient particulièrement adaptées aux problèmes d'optimisation multimodale en permettant une exploration plus efficace de l'espace des solutions dans le cas de problème d'optimisation multimodale.

La définition des méthodes de scores (cf. section 4.3) nous a permis de définir des fonctions d'évaluation permettant la sélection de modèles dans l'espace des structures. Il s'avère que le problème de la sélection de la meilleure structure (ou, tout du moins, d'une structure de bonne qualité) s'avère difficile. Les fonctions d'évaluation, qu'elles tiennent compte ou non de la complexité du modèle évalué – comme le score BIC –, présentent de nombreux optima locaux.

Les méthodes de *niching* paraissent donc être un choix adéquat pour le traitement de l'apprentissage de structures.

Cependant, les méthodes de *niching* se divisent en deux catégories : les méthodes dites spatiales et les méthodes dites temporelles. Ces deux familles de méthodes ont pour point

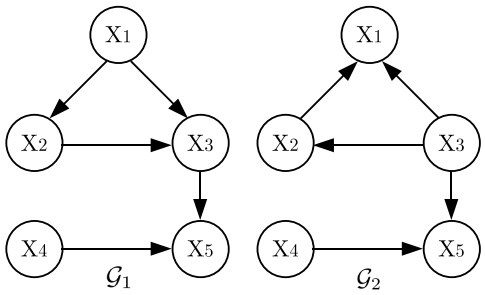
commun la définition d'une notion de distance permettant de définir et maintenir les différentes niches dans lesquelles seront réparties, idéalement, les différents individus. Avant de définir quelle stratégie de *niching* nous allons adopter, nous commencerons par essayer de définir une distance employable dans l'espace des solutions à notre problème.

6.2.1 Distances entre deux structures de réseaux bayésiens

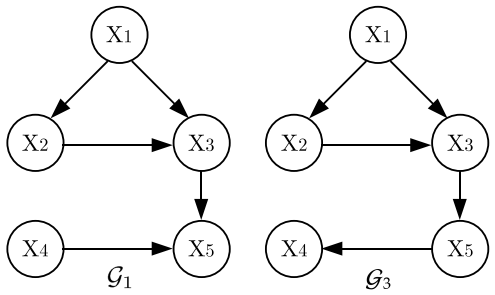
Les premières méthodes de *niching*, appliquées à des solutions encodées dans l'espace binaire, définissaient une distance sur l'espace génotypique par le biais d'une distance de Hamming. Cette notion a-t-elle un sens dans le cadre de l'apprentissage de structures ?

Si l'on se réfère à la notion de graphes équivalents au sens de Markov, celle-ci implique que deux structures différentes peuvent proposer la même décomposition de la loi jointe sur la domaine – et donc obtenir le même score, si ce dernier est équivalent (cf. section 4.3) –. Cela signifie qu'une méthode de *niching* employant une distance basée sur une distance de Hamming dans l'espace des structures considérera, par exemple, les deux structures $\mathcal{GE}_1$ et $\mathcal{GE}_2$ de la figure 6.5-a (ou deux structures présentant un nombre conséquent de différences similaires) comme dissociées alors qu'elles appartiennent toutes deux à la classe d'équivalence représentée par le graphe essentiel $\mathcal{GE}_1$ de la figure 6.5-c.

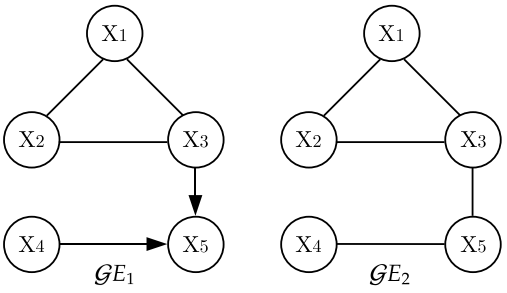
Inversement, les *GOSC* $\mathcal{G}_\infty$ et $\mathcal{G}_\ni$, figure 6.5-b, dont les classes d'équivalences sont respectivement représentées par les *GE* $\mathcal{GE}_1$ et $\mathcal{GE}_2$ de la figure 6.5-c – seront étiquetés comme étant plus proches.



(a) Structures équivalentes présentant une distance de Hamming de 3.



(b) Structures non-équivalentes présentant une distance de Hamming de 1.



(c) Graphes essentiels $\mathcal{GE}_1$ et $\mathcal{GE}_2$.

Figure 6.5 – Exemples de l'application de la distance de Hamming dans l'espace des structures.

Une distance de Hamming définie sur l'espace des structures ne parait donc pas adéquate car elle ne tient pas compte de propriétés essentielles des structures considérées.

[Mahfoud, 1995] recommande l'emploi de distances définies sur l'espace phénotypique. Une telle distance, dans le cas des structures de réseaux bayésiens, pourrait être la divergence de Kullback-Leibler (cf. annexe A) ou une autre mesure de divergence basée sur l'entropie [Lin, 1991]. La divergence de Kullback-Leibler, sous une forme adaptée, permet de définir une distance entre deux réseaux bayésiens (plus exactement, entre les deux distributions de probabilités de la loi jointe représentées par ces réseaux bayésiens). Cependant, la divergence de Kullback-Leibler présente un inconvénient majeur : la quantité de calculs requise. Il en effet nécessaire, pour calculer cette distance, de calculer un logarithme sur l'ensemble des instanciations du domaine présentes dans la base d'apprentissage. Pour un réseau tel que le réseau INSURANCE présenté dans le chapitre 8, préalablement aux résultats expérimentaux, il est alors nécessaire de procéder à l'évaluation de nombreuses probabilités jointes. Sachant que les distances, dans une méthode de *niching* doivent être calculées rapidement pour les différents individus, la quantité de calculs requise par la divergence de Kullback-Leibler rend l'emploi de cette dernière rédhibitoire. De plus, même en ignorant la complexité de calcul de la divergence, deux critères viennent définitivement interdire l'emploi de celle-ci dans une méthode de *niching* : la divergence de Kullback-Leibler n'est pas symétrique et ne respecte pas l'inégalité triangulaire, interdisant ainsi son emploi en tant que distance (nous verrons qu'une divergence – de Jensen-Shannon – symétrique, bornée et respectant l'inégalité triangulaire nécessite encore plus de calculs).

Concrètement, le problème de la détermination d'une distance entre deux structures candidates pour un même domaine de variables demeure un problème ouvert. La littérature consacrée à l'apprentissage de structures de réseaux bayésiens emploie, dans l'évaluation de la qualité des résultats d'une méthode, diverses mesures (écart de score, distance de Hamming, etc.) en conjugaison, aucun mesure ne pouvant à elle seule permettre la détermination d'une réelle différence entre deux structures.

Malgré cela, nous avons mentionné une propriété importante des structures de réseaux bayésiens, propriété permettant de regrouper les structures proposant des modélisations similaires : la notion de graphes équivalents. Deux graphes structurellement différents (par certaines orientations) peuvent encoder une même décomposition de la loi jointe sur le domaine modélisé. L'emploi des classes d'équivalence en tant que niche, au sein d'une stratégie de *niching* appliquée à l'apprentissage de structures paraît donc être une possibilité.

Un autre point motivant l'emploi des classes d'équivalence est que la détermination d'une méthode de *niching* adaptée passe par la détermination d'un paramètre supplémentaire : le rayon d'un niche :

- un rayon trop grand risque d'affecter d'éventuels optima proches dans l'espace employé (génotypique ou phénotypique) ;
- si le rayon est trop faible, l'algorithme perd vraisemblablement en efficacité.

En général, le rayon employé est déterminé soit par une connaissance *a priori* du problème ou par détermination empirique, ce qui rend la détermination d'un rayon adéquat difficile, en pratique. Le choix des classes d'équivalence en tant que niches permet de s'affranchir de ces problèmes en négligeant la notion de distance.

La détermination de l'appartenance ou non à une même classe d'équivalence – et donc l'appartenance de deux individus à une même niche – peut se faire simplement, par le biais de la

distance de Hamming calculée entre les graphes essentiels représentant les classes d'équivalences respectives des deux structures, plus précisément à partir des matrices d'adjacence des graphes essentiels correspondants – l'appartenance à la même classe d'équivalence impliquant alors que la distance de Hamming entre les *GE* est nulle –. L'obtention du graphe essentiel d'une structure – et donc la caractérisation d'une niche – peut se faire alors rapidement en employant, par exemple, les algorithmes de calcul des *GE* utilisés par [Chickering, 2002b].

Le nombre de *GOSC* différents appartenant à une même classe d'équivalence est limité – 3,7 d'après [Perlman et Gillispie, 2001] –. Par conséquent, on peut légitimement remettre en question la pertinence d'un tel choix. Il faut cependant se replacer dans le cadre de notre problématique. Il existe en effet de nombreux optima locaux en l'espace des *GOSC* lorsque nous employons un méthode de score classique telle que les scores BDeu ou BIC ; en revanche, nous pouvons légitimement penser que les structures obtenant un score élevé et correspondant à des optima locaux pour la fonction d'évaluation employée sont structurellement assez proches de l'optimum global recherché.

Par la suite, nous définissons donc une niche comme l'ensemble des *GOSC* appartenant à un même classe d'équivalence au sens de Markov. La définition d'une niche étant effectuée, il nous reste à éliciter le type de méthode de *niching* au sein de laquelle l'employer.

6.2.2 Choix d'une méthode d'optimisation

Le choix d'une méthode de *niching* implique en premier lieu de choisir entre une approche temporelle (*niching* séquentiel) ou spatiale (méthodes de *sharing* ou de *crowding*).

Le consensus général, exprimé dans [Mahfoud, 1995], est que les méthodes relevant du *niching* spatial renvoient de meilleurs résultats que le *niching* séquentiel. Mahfoud explique ces différences de performances par les observations suivantes :

1. le *niching* séquentiel, à travers son système de dégradation séquentielle de la valeur de la *fitness* aux points optimaux, modifie le paysage de la fonction d'évaluation et, consécutivement :
   - de faux optima risquent d'apparaître à la limite des zones dégradées,
   - l'exploration de l'espace de recherche se voit entravée par la création de zones de faible *fitness* ; l'exploration et donc l'apparition de matériel génétique correspondant aux individus de ces zones est découragée,
   - d'éventuels optima, situés à proximité d'optima détectés – en deçà du rayon de la niche correspondante – sont ignorés car leur *fitness* est arbitrairement dégradée. Ceci étant essentiellement dû à une mauvaise définition de la taille des niches créées,
2. on peut observer un phénomène de convergence répétée vers les zones précédemment dépréciées.

Nous pouvons néanmoins remarquer que si les principaux reproches émis à l'égard du *niching* séquentiel concernent la modification de la *fitness*, ce comportement est partagé par les méthodes spatiales de type *sharing*. De plus, un des reproches évoqués, à savoir la destruction éventuelle d'optima locaux au voisinage des optima détectés, ne saurait être pris en compte dans notre cas. En effet, si nous définissons une niche comme consistant en l'ensemble des

structures appartenant à une même classe d'équivalence au sens de Markov, la pénalisation de la *fitness* de ces structures ne saurait modifier l'évaluation de celles appartenant à d'autres classes d'équivalence, voisines dans l'espace des solutions.

Le *niching* séquentiel présente cependant l'avantage de sa simplicité d'implémentation ; celle-ci est plus aisée et sensiblement plus intuitive que celle d'une méthode spatiale puisqu'elle consiste seulement en l'ajout d'une méthode de pénalisation/mémorisation à une méthode évolutionnaire prédéfinie.

Si l'avantage théorique semble bien revenir aux méthodes spatiales, nous avons entrepris dans un premier temps d'implémenter une méthode de *niching* séquentiel à la recherche de structures. Ce choix s'explique en premier lieu par le souhait d'éprouver une méthode relativement simple à mettre en place. De plus, nous le verrons par la suite, au regard de certains développements récents dans le domaine des méta-heuristiques, une approche intéressante consiste à non plus choisir entre les approches spatiales et temporelles mais à conjuguer leurs propriétés à travers une hybridation. Nous étudierons cette dernière approche dans la suite de ce chapitre.

6.2.3 *Niching* séquentiel appliqué à l'apprentissage de structures

L'algorithme procède à une évolution comparable à celle d'un algorithme génétique classique (cycles itérés d'évaluation, sélection, reproduction et remplacement des individus) à ceci près qu'une liste d'optima est tenue à jour. Les individus correspondant à ces optima voient leur *fitness* dégradée afin de décourager toute visite et maintenance de ces individus.

6.2.3.1 Optima locaux

Comme nous l'avons précisé dans la section, les optima locaux, dans le cadre de notre méthode, correspondent à des classes d'équivalence au sens de Markov. Quand au moins une classe d'équivalence a été étiquetée comme correspondant à un optimum de la *fitness*, les différents individus de la population appartenant à un optimum de cette liste voient la valeur de leur *fitness* dégradée afin de décourager toute exploitation ultérieure de ces parties de l'espace des solutions. La détermination de l'appartenance ou non d'un individu à une classe d'équivalence de la liste intervient lors de la phase d'évaluation, après génération par croisement et mutation de la nouvelle population. Le *GE* de chaque nouvel individu est alors calculé et comparé à ceux contenus dans la liste des optima. Si une correspondance est déterminée, alors l'individu concerné voit sa *fitness* pénalisée et fixée à une valeur arbitraire (très faible, inférieure au score de la structure vide).

Les classes d'équivalence répertoriées par la liste sont déterminées au cours du déroulement de l'algorithme : si, au terme d'un nombre prédéterminé Ite_{opt} d'itérations, il n'y a pas d'amélioration de la *fitness* du meilleur individu, l'algorithme récupère le *GE* de la classe d'équivalence de celui-ci et l'ajoute à la liste.

Il est important de remarquer ici que les optima locaux ne sont pas interdits de manière formelle dans la population. Les optima enregistrés peuvent très bien réapparaître dans notre

population suite à un croisement. L'évaluation des classes d'équivalences présentes ne commence en effet qu'à l'issue de la phase de mutation ; un optimum préalablement mémorisé peut fort bien réapparaître à l'issue de l'opération de croisement et l'individu concerné subir une mutation permettant d'explorer le voisinage de cet optimum.

6.2.3.2 Comportement de l'algorithme

Les auteurs de [Beasley et al., 1993] procèdent à une réinitialisation du processus évolutif après chaque détermination d'un optimum. Notre algorithme poursuit l'évolution en considérant la liste, mise à jour, de ces optima. Néanmoins, en permettant à la population de poursuivre son évolution au voisinage des optima détectés, nous cherchons à préserver les différentes briques élémentaires jusqu'alors trouvées ainsi qu'à réduire le nombre d'évaluations que requerraient plusieurs lancements de l'algorithme.

6.2.3.3 Arrêt de l'algorithme

À la rencontre d'un critère d'arrêt, l'algorithme génétique termine donc son exécution en renvoyant la liste des optima déterminés jusqu'alors. Le critère d'arrêt de l'algorithme peut aussi être envisagé de manières différentes, par exemple :

- soit après un nombre fixé d'optima locaux détectés ;
- soit après un nombre fixé d'itérations (générations) en tout.

Nous optons pour la deuxième possibilité. Choisir un nombre fixe d'optima locaux peut, en effet, s'avérer être un choix nettement plus arbitraire que celui du nombre d'itérations. En fonction du problème considéré et/ou des données d'apprentissage, le nombre d'optima locaux en lesquels le processus évolutif – ou une autre heuristique de parcours – peut stagner peut varier.

À terme, l'algorithme renvoie un *GOSC* correspondant à l'instanciation du *GE* rattaché au meilleur score au sein de la liste des optima.

Le fonctionnement de l'algorithme est donné, en détail, dans le pseudo-code de l'algorithme 11.

Un paramètre important de l'algorithme est, à première vue, le seuil au delà duquel un individu est identifié en tant qu'optimum de la fonction d'évaluation.

Il est nécessaire de définir une valeur de ce paramètre, que nous nommons Ite_{opt}, qui soit :

- ni trop petite : considérer trop hâtivement une classe d'équivalence comme étant un optimum local entraverait l'exploration de l'espace de recherche de l'algorithme génétique et l'on cumulerait de plus un trop grand nombre d'optima ;
- ni trop grande : perte du bénéfice de la méthode en restant trop longtemps en un même point de l'espace de recherche : les optima locaux freinent alors effectivement la progression de la recherche.

L'expérience nous a montré qu'une valeur de Ite_{opt} située entre 15 et 25 itérations permet d'obtenir de bons résultats. La méthode semble en effet assez stable au niveau de la valeur du paramètre Ite_{opt} tant que, comme nous l'avons évoquée, cette valeur permet à la fois de rester

Algorithme 11 Algorithme génétique avec mémorisation des optima rencontrés.

Entrée: Taille de la population λ, nombre d'itérations avant mémorisation Ite_{opt}, score pénalisé *Penal*, probabilité de mutation P_{mute} et probabilité de croisement P_{cross}.

Sortie: Liste *optima* des optima détectés sous forme des *GE* représentant les meilleures classes d'équivalence au sens de Markov trouvées et $\mathcal{G}_{eq}^{k}$, meilleure classe d'équivalence au sens du score de *optima*.

1: /***Création de la population initiale Pop^0 par MWST***/
2: *compteur* $\leftarrow 0$
3: *optima* $\leftarrow \{\}$
4: **Pour** *Ite* itérations **Faire**
5: /***Phase de sélection***/
6: /***Phase de croisement***/
7: /***Phase de mutation***/
8: *récupération de la population* Pop'^t *générée par les opérateurs génotypiques*
9: **Si** *optima* $\neq \{\}$ **Alors**
10: **Pour** $i = 1 : \lambda$ **Faire**
11: **Pour** $j = 1 : |optima|$ **Faire**
12: **Si** le *GE* de $Pop'^t(i) = optima(j)$ **Alors**
13: $Score(Pop'^t(i)) \leftarrow Penal$
14: **Fin Si**
15: **Fin Pour**
16: **Fin Pour**
17: **Fin Si**
18: **Si** $compteur = Ite_{opt}$ **Alors**
19: *optima* $\leftarrow$ le *GE* de $(Pop'^t(1))$
20: Identifier tous les individus $Pop'^t(i) = \{\mathcal{G}_i, Score(\mathcal{G}_i)\}, i \in \{2, \dots \lambda\}, \mathcal{G}_i \equiv \mathcal{G}_1$
21: $Score(Pop'^t(i)) \leftarrow Penal$
22: *compteur* $\leftarrow 0$
23: **Sinon**
24: *compteur* $\leftarrow$ *compteur* $+ 1$
25: **Fin Si**
26: /***Génération de* Pop^{t+1}***/
27: **Fin Pour**
28: Retourner $\mathcal{G}_{eq}^{(k)}$ avec $k = argmax(Score($ l'instanciation en *GOSC* de $(\mathcal{G}_{eq}^{(k)})), \forall \mathcal{G}_{eq}^{(k)} \in optima$

peu de temps autour d'un même optimum tout en permettant à la population de converger autour de celui-ci.

La valeur de la pénalité infligée aux classes d'équivalences est, nous l'avons dit, arbitraire. La seule contrainte est que la valeur à laquelle est abaissée l'évaluation de l'optimum détecté soit inférieure à celle de la plus mauvaise structure possible, par exemple : -10^{15}.

6.2.4 Expérimentations et résultats

Nous avons éprouvé, dans un souci de comparaison de performances, plusieurs méthodes dont :

- l'algorithme génétique simple, défini en début de ce chapitre ;
- l'algorithme génétique appliquant la stratégie de *niching* séquentiel ;
- certaines des principales méthodes de la littérature présentées dans le chapitre 4 ;

Les résultats de ces expérimentations ont été regroupés dans le chapitre 8. Comme nous l'avons fait remarquer, bien qu'une méthode de *niching* séquentiel paraisse *a priori* plus adaptée à notre problématique qu'une méthode spatiale tel que le *sharing* ou le *crowding*, [Mahfoud, 1995] souligne les avantages et performances des méthodes spatiales. Des travaux récents [Zaharie, 2004, Zhang et al., 2006] montrent qu'il est cependant possible de combiner une approche temporelle et une approche spatiale. Ce type d'hybridation présente plusieurs avantages et nous avons décidé d'en implémenter une version, présentée dans la suite de ce chapitre, employant le mécanisme séquentiel présenté dans cette section.

6.3 Combinaison avec une approche spatiale

Bien que nous ayons choisi d'employer une méthode de *niching* séquentiel, nous pouvons légitimement nous interroger sur les avantages que pourrait avoir, pour la résolution de notre problème, l'application d'une stratégie de répartition spatiale. Nous avons en effet vu qu'une approche prônant une répartition spatiale des individus présentait, elle aussi, des avantages, notamment dans la découverte et la perpétuation d'un matériel génétique diversifié.

Un comparatif entre une méthode spatiale telle que le *sharing* ou le *crowding* (cf. section 5.4.3) avec une méthode séquentielle n'est cependant pas l'objet de ce travail de thèse ; par conséquent, nous avons décidé de mettre au point et d'implémenter une méthode permettant de combiner les deux aspects (temporel et spatial) des méthodes de *niching*.

Pour cela, nous avons combiné les notions vues dans la section précédente à une technique de répartition de la population telle qu'appliquée dans le cadre des algorithmes génétiques parallèles (cf. section 5.4.4). Nous décrivons dans la suite comment s'opère cette hybridation, quelle est la stratégie de répartition des individus ainsi que l'implémentation finale de l'algorithme combinant les deux notions.

6.3.1 Répartition spatiale de la population

Il existe plusieurs variantes pour l'implémentation d'algorithmes distribués. Parmi celles-ci, nous avons précédemment abordé le principe du modèle en îlots (ou *island model*). Les théories communément associées aux modèles distribués et plus particulièrement aux modèles en îlots sont essentiellement issues de travaux concernant la génétique des populations et visent à expliquer les phénomènes de diversification et de transmission de gènes au sein de populations distribuées.

6.3.1.1 Génétique des populations

La théorie des algorithmes génétiques distribués est fréquemment reliée aux théories de la génétique des populations. Nous n'entrerons cependant pas dans le détail des considérations de cette branche de recherche et nous nous contenterons de citer les principaux points en relation avec l'algorithmique évolutionnaire.

Un des modèles de système parallèle les plus implémentés est le modèle en îlots (ou *Island model*). Ce terme est en fait à rapprocher de son vis-à-vis dans le domaine de l'étude génétique : le modèle de populations en îles de Wright [Wright, 1964] (le même S. Wright auquel est communément attribuée la paternité des réseaux bayésiens [Wright, 1921]).

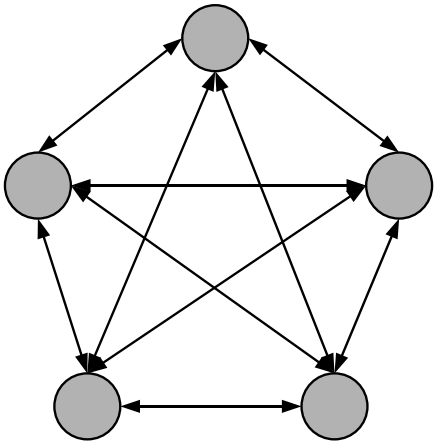


Figure 6.6 – Modèle de populations en îlots. Ici, les populations sont complétement interconnectées.

Ce modèle consiste en un ensemble de d populations localisées chacune sur des îles indépendantes (voir figure 6.6). Les îlots (on utilise, en génétique, le terme *dèmes*) peuvent échanger des membres de leur population avec d'autres dèmes. Notons que ce modèle est essentiellement théorique et ne sert, en biologie, qu'afin de pouvoir modéliser aisément les phénomènes de migration et d'échanges de gènes entre populations.

Il existe, localement à chaque population, un phénomène de dérive génétique, amenant les différents génomes vers un stade d'homogénéité des génomes. Ces différents phénomènes de dérive sont supposément inter-indépendants. Du fait de l'introduction du phénomène de

migration, ces dérives perdent leur indépendance dans chaque population, amenant l'introduction de nouveaux allèles dans des dèmes où un autre allèle s'était fixé. Sans aller plus loin dans le domaine biologique, nous pouvons d'ores et déjà établir la conséquence du phénomène migratoire dans cette modélisation : le taux d'individu partageant le même matériel génétique au sein d'une sous population ne va plus tendre vers 1 comme cela est le cas avec un algorithme panmictique.

Cette homogénéité diminue d'autant plus avec la croissance du nombre ponctuel de migrants arrivant à chaque génération. La conclusion, en termes d'algorithmique évolutionnaire, est que la diversité génétique est plus grande au sein d'une population subdivisée qu'au sein d'une population panmictique.

D'autres modèles de populations distribués existent, mais leur différence réside dans des modes de transitions et de définitions différentes de voisinage et, surtout, les implémentations de ces modèles pour l'algorithmique évolutionnaire se sont avérées infructueuses.

Une autre théorie, issue elle aussi de la biologie et fréquemment acceptée comme une explication aux performance des algorithmes distribués, est celle de l'équilibre intermittent (ou *punctuated equilibrium*). D'après cette théorie [Eldredge et Gould, 1972, Martin et al., 1997], l'évolution des espèces est caractérisée par de longues périodes de stabilité entrecoupées de courtes mais remarquables phases de changements. Dans un algorithme génétique, les périodes de stabilité correspondent à une convergence prématurée. Lorsque plusieurs sous populations évoluent en parallèle, les migrations ponctuelles d'une sous population à une autre permettent l'injection de nouveau matériel génétique et par conséquent un redémarrage de l'étape d'exploration de l'espace des solutions.

6.3.1.2 Notre implémentation

Bien que les modèles en îlots, tels qu'ils sont introduits dans la section 5.4.4, s'inscrivent dans le cadre d'implémentations réparties sur plusieurs processeurs. Nous avons choisi ici d'implémenter notre méthode sur un unique processeur. L'implémentation sur plusieurs machines en parallèle est bien entendu tout à fait envisageable, néanmoins nous nous contenterons dans le cadre de nos travaux d'évaluer notre implémentation en termes de performances des individus évalués, critère indépendant de la nature de l'implémentation.

Typiquement, pour un modèle de populations réparties en îlots, des paramètres additionnels entrent en ligne de compte, en plus des paramètres d'un algorithme panmictique [Tanese, 1989] :

l'intervalle migratoire : noté I_{mig}, il s'agit du nombre d'itérations au sein d'une ou de l'ensemble des sous-populations considérées séparant deux phases migratoires ;

le taux de migration : noté T_{mig}, indique le taux d'individu, au sein d'une sous-population, sélectionnés pour la migration ;

nombre de sous populations : le nombre d'îlots entre lesquels s'opéreront les différentes migrations ;

la taille des sous populations : communément, des populations de tailles identiques évoluent sur les différents îlots. La taille de chaque population est égale à une fraction du nombre total d'individus. Cependant, rien n'empêche de généraliser en proposant des tailles distinctes pour chaque population.

Si l'on envisage de combiner la répartition spatiale avec la méthode de *niching* séquentiel présentée dans le chapitre précédent, l'algorithme s'apprête tout naturellement à l'ajout d'une mémorisation des optima locaux rencontrés par les différentes populations.

Le fonctionnement de l'algorithme est alors le suivant :
- chaque sous-population évolue pendant I_{mig} itérations avant de transférer $T_{mig} \times \lambda$ individus où λ indique la taille commune de chaque sous-population ;
- chaque sous-population détecte, localement, d'éventuels optima selon le principe énoncé dans le chapitre 6 ;
- une liste des optima mémorisés, commune à l'ensemble des sous-populations est tenue à jour.

L'intérêt d'un modèle en îlots est la mise en place du système de migration ponctuelle précédemment évoqué et permettant un échange de briques élémentaires entre les différentes sous-populations. En l'absence de cette interactivité, nous ne procéderions qu'à N_{pop} instances d'un même algorithme génétique, où N_{pop} est le nombre de sous-populations évoluées. À cette interactivité vient donc s'ajouter l'emploi d'une liste commune d'optima locaux, lesquels voient la valeur de leur *fitness* dépréciée.

L'objectif final d'une telle implémentation étant alors de parvenir à une répartition maximale dans l'espace des solutions des différentes sous-populations évoluées.

Une description plus détaillée de notre implémentation est fournie par l'algorithme 12.

6.3.2 Expérimentations et résultats

Les résultats des tests effectués à partir des deux méthodes présentées dans ce chapitre ont été regroupés dans le chapitre 8.

Les tests préliminaires effectués avec notre méthode hybride – présentés en annexe C – ont montré une stabilité du comportement de l'algorithme vis-à-vis des paramètres de migration. Nous avons néanmoins opté pour un réglage de ces paramètres à 20 itérations pour l'intervalle migratoire et 10% de la population pour le taux de migration. Le nombre de sous populations ainsi que la taille en nombre d'individus de ces dernières ont été tout deux fixés à 30.

Algorithme 12 Algorithme génétique distribué avec mémoire

Entrée: Taille des populations λ, nombre de populations N_{pop}, nombre d'itérations avant mémorisation Ite_{opt}, pénalité $Penal$, probabilité de mutation P_{mute} et probabilité de croisement P_{cross}, taux de migration T_{mig}, intervalle migratoire I_{mig}, nombre total d'itérations N_{ite}.

Sortie: Liste $Optima$ des optima détectés, graphe partiellement orienté $\mathcal{G}_{eq}$ représentant la meilleure classe d'équivalence au sens de Markov trouvée.

/***Création des populations initiales $Pop_k^0, k \in \{1, \dots N_{pop}\}$***/
$compteur_k \leftarrow 0$, $\forall k \in \{1, \dots N_{pop}\}$
$Optima \leftarrow \{\}$
$optima_k \leftarrow \{\}$, $\forall k \in \{1, \dots N_{pop}\}$
Pour N_{ite} itérations **Faire**
  Pour chacune des N_{pop} populations, notée Pop_k, $k \in 1, \dots N_{pop}$ **Faire**
    Pour I_{mig} itérations **Faire**
      $optima_k \leftarrow Optima$
      /***Phase de sélection***/
      /***Phase de croisement***/
      /***Phase de mutation***/
      récupération de la population $Pop_k'^t$ générée par les opérateurs génotypiques
      Si $optima_k \neq \{\}$ **Alors**
        Pour $i = 1 : \lambda$ **Faire**
          Pour $j = 1 : |optima_k|$ **Faire**
            Si le GE de $(Pop_k'^t(i)) = optima_k(j)$ **Alors**
              $Score(Pop_k'^t(i)) \leftarrow Penal$
            Fin Si
          Fin Pour
        Fin Pour
      Fin Si
      Si $compteur_k = Ite_{opt}$ **Alors**
        $optima_k \leftarrow$ le GE de $(Pop_k'^t(1))$
        Identifier tous les individus $Pop_k'^t(i) = \{\mathcal{G}_i, Score(\mathcal{G}_i)\}, i \in \{2, \dots \lambda\}, \mathcal{G}_i \equiv \mathcal{G}_1$
        $Score(Pop_k'^t(i)) \leftarrow Penal$
        $compteur_k \leftarrow 0$
      Sinon
        $compteur_k \leftarrow compteur_k + 1$
      Fin Si
      /***Génération de Pop_k^{t+1}***/
      $Optima \leftarrow optima_k$
    Fin Pour
  Fin Pour
  /***Phase migratoire***/
  Pour $k = 1, \dots, N_{pop}$ **Faire**
    Les $\left\lfloor T_{mig} \times \lambda \right\rfloor$ meilleurs individus de Pop_k migrent vers une population Pop_j, $j \neq k$, choisie aléatoirement
  Fin Pour
Fin Pour
Retourner $\mathcal{G}_{eq}^{(i)}$ avec $i = argmax(Score($l'instanciation en $GOSC$ de$(\mathcal{G}_{eq}^{(i)})), \forall \mathcal{G}_{eq}^{(i)} \in Optima$

Chapitre 7

Stratégie d'adaptation de la mutation

Le milieu naturel, en tant que processus évolutif, a pour particularité d'apprendre de ses actions passées ; lorsqu'un individu ou un comportement disparait consécutivement à de mauvaises performances, il est rare que celui-ci réapparaisse ultérieurement. Alors que les algorithmes évolutionnaires tendent effectivement à promouvoir l'émergence et la survie des meilleurs individus, leur nature stochastique même a pour inconvénient de générer nombre de calculs inutiles en générant plusieurs fois des solutions de mauvaise qualité ayant déjà été explorées.

Nous introduisons dans ce chapitre une méthode ayant pour objectif de permettre une adaptation de l'exploration de l'espace des structures en fonction des solutions précédemment visitées et évaluées. Cette adaptation ayant pour but d'une part d'inciter l'apparition d'éléments performants et, d'autre part, de décourager la génération d'un matériel génétique de mauvaise qualité.

Dans une première partie, nous aborderons le principe général de notre méthode ainsi ses motivations. La deuxième partie de ce chapitre s'attachera à une description précise de notre implémentation.

7.1 Introduction

L'objectif d'un algorithme génétique est de permettre, idéalement (opérateurs adéquats et utilisation de bons paramètres), de couvrir efficacement l'espace des solutions et de dégager la ou les partie(s) de cet espace contenant de "bonnes" solutions.

Dans la réalité, ces méthodes sont employées pour la résolution de problèmes comportant un espace de solutions de taille conséquente et difficile à explorer. Dans ce cas, l'approche *generate and test* des méthodes évolutionnaires a alors pour conséquence une convergence lente de l'algorithme vers une bonne solution et celui-ci effectue inutilement de nombreux calculs liés à l'évaluation d'individus non-optimaux, eux-mêmes souvent explorés à plusieurs reprises.

Une approche consiste dès lors à orienter l'évolution de la population en tenant compte des résultats des individus précédemment évalués afin, d'une part, d'encourager l'exploration

et l'exploitation de secteurs intéressants de l'espace de recherche et, d'autre part, d'éviter les évaluations redondantes de génotypes de mauvaise qualité. Ce type de comportement (adaptivité en fonction d'un retour sur la qualité des individus) est ouvertement abordé par deux types de méthodologies évolutionnaires : les approches adaptatives (cf section 5.4.1) et les approches de type EDA (cf section 5.4.2).

Les méthodes adaptatives visent à optimiser les opérateurs génétiques et/ou les paramètres associés à ces derniers soit par un retour direct sur la qualité des individus évolués, soit en laissant au processus évolutif le soin de favoriser la survie, conjointement aux individus associés, des paramètres adaptés. Les méthodes d'estimation de distribution, telles que l'algorithme *PBIL* (*Population Based Incremental Learning*) [Baluja, 1994], permettent de faire évoluer une densité de distribution évaluée en fonction des meilleurs individus rencontrés jusqu'alors.

D'autres méthodes implémentent cette idée de mémorisation au sein des opérateurs génotypiques d'un algorithme évolutionnaire, favorisant la génération d'individus de meilleure qualité [Sebag et al., 1998] où l'action de l'opérateur de mutation est influencée par une stratégie préétablie ainsi que des caractéristiques des meilleurs et pires individus rencontrés.

Une première version [Delaplace et al., 2006], auto-adaptative, de l'approche décrite dans ce chapitre visait à déterminer les probabilités de mutation appliquées localement aux ensembles de sommets parents des différentes variables d'une structure. Le principe général de cette approche était alors de promouvoir les mutations d'individus dégradés afin de favoriser l'exploration de l'espace des solutions tout en réduisant les mutations d'individus de meilleure qualité, proches d'un optimum. Cette approche a cependant l'inconvénient de ne considérer, pour un ensemble de sommets parents et pour chaque occurrence d'une opération de mutation sur cet ensemble, d'influer sur la probabilité de l'ensemble des opérations de mutation. Le problème étant que, parmi les mutations affectées, certaines peuvent avoir des conséquences différentes sur la qualité de la solution.

Consécutivement à ces observations, une meilleure approche permettrait :
- de pouvoir évaluer les résultats des différentes opérations de mutation en fonction de leur influence *individuelle* sur le score global de la structure,
- d'influer consécutivement sur la probabilité de survenance de ces mouvements dans l'ensemble de la population.

C'est dans cette optique que nous avons développé une méthode adaptative permettant d'orienter l'exploration de l'espace des solutions en fonction d'un retour sur la qualité des explorations passées. La section suivante décrit les éléments théoriques de cette méthode.

7.2 Notre méthode

Dans un premier temps, nous rappelons le fonctionnement de l'opérateur de mutation dans l'algorithme génétique simple, défini au chapitre précédent.

L'opérateur de mutation employé dans notre moteur évolutionnaire définit une mutation comme une opération parmi :
- ajout d'un arc ;

– inversion d'un arc ;
– soustraction d'un arc.

Cette opération est appliquée, avec une probabilité P_{mute}, à un arc a_{ij} et modifie ainsi l'ensemble Π_j des sommets parents du sommet X_j. La nature de l'opération ainsi que celle de l'arc a_{ij} sont effectués séquentiellement et de manière uniformément aléatoire en fonction de la possibilité, ou non, d'appliquer les différentes opérations – un ensemble de sommets $\Pi_j = \emptyset$ ne peut se voir appliquer qu'un ajout d'arc, aucun arc ne pouvant être soustrait ou inversé –.

Nous souhaitons définir une distribution de probabilités permettant d'influer sur le choix des opérations de mutation effectuées sur les individus de la population. Cette distribution devant refléter les résultats des mutations passées en fonction de l'impact de celles-ci sur la qualité des individus modifiés.

Notons que bien que nous employons une distribution de probabilités évoluant dans le temps, notre méthode se distingue d'une méthode de type EDA en ce que nous ne faisons pas évoluer une distribution de probabilités régissant les individus mêmes mais l'orientation de l'exploration. À proprement parler, nous faisons bel et bien évoluer une population d'individus. La distribution de probabilités définie sur les opérations de mutation est, quand à elle, modifiée par rapport à ceux-ci.

Considérons la probabilité, lors d'une phase de mutation, que le coefficient $a_{ij} \in \{0, 1\}, \quad 1 \leq i, j \leq n$, représentant l'existence (ou l'absence) de l'arc $X_i \rightarrow X_j$, subisse une opération Op_{mute} parmi $\{ajout, inversion, soustraction\}$. Nous écrivons cette probabilité $P(i, j, Op_{mute})$.

Influencer les probabilités d'application des différentes opérations de mutation revient alors à mettre au point un mécanisme de contrôle de $P(i, j, Op_{mute})$. Il nous reste à définir ce mécanisme.

Commençons par considérer que nous manipulons directement la distribution $P(i, j, Op_{mute})$, pour tout couple $1 \leq (i, j) \leq n, i \neq j$ et pour toute opération Op_{mute}. Nous devrions alors définir et maintenir $3 \times n \times (n-1) - 1$ paramètres indépendants. Il apparaît rapidement que, dès lors que le réseau dont nous cherchons la structure contient un nombre élevé de variables, les probabilités ainsi définies seront vraisemblablement faibles et leur variation peu influente.

L'opération de mutation se décompose suivant le choix de i,j et Op_{mute}. Nous pouvons simplifier la densité de probabilités en conditionnant un sous ensemble de $\{i, j, x\}$ par son complémentaire, lequel sera sélectionné par le biais d'une distribution de probabilités statique. Nous allons étudier les sous-ensembles possibles et choisir en conséquence.

Les distributions de probabilités que nous pouvons définir et contrôler peuvent être :

$P(Op_{mute}, i|j)$: nous déterminons l'opération effectuée et le sommet d'arrivée de l'arc concerné en fonction du sommet parent. L'inconvénient est que le choix de Op_{mute}, étant donné j, est fortement déterminé par celui de i (le seul choix survient entre *inversion* et *soustraction*, lorsque l'arc a_{ij} existe. L'intérêt du contrôle de cette distribution est trop limité pour nous intéresser ;

$P(i, j|Op_{mute})$: l'arc sur lequel s'applique l'opération dépend de la nature de celle-ci. Ce choix implique de devoir choisir entre
  – laisser l'opérateur de mutation intervenir plus d'une fois sur un même ensemble de parents et, donc, devoir réestimer la distribution $P(i, j|Op_{mute})$ entre chaque opération ;

– interdire à l'opérateur de mutation d'intervenir plus d'une fois sur un même ensemble Π_k et devoir réestimer la distribution $P(i, j|Op_{mute}), j \neq k$ entre chaque opération de mutation ;

$P(Op_{mute}|i, j)$: ici, l'opération effectuée est déterminée par le choix préalable de l'arc. Ce choix n'a pas d'intérêt car la détermination de l'arc a_{ij} sur lequel opérer implique, au plus, de choisir entre une opération de soustraction ou d'inversion si $a_{ij} = 1$. Dans le cas contraire, seule l'opération d'ajout est possible. ;

$P(i|Op_{mute}, j)$: revient à déterminer le sommet de départ de l'arc en fonction de l'opération élicitée. Le sommet d'arrivée étant déterminé, nous pouvons alors définir n distributions $P(i|Op_{mute}, j), i \neq j, 1 \leq i, j \leq n$ comptant chacune $n - 2$ paramètres indépendants. Chaque ensemble de sommets parents des variables du domaine peut alors muter une seule fois avec la probabilité P_{mute} et il est inutile de procéder à des calculs intermédiaires entre chaque opération ;

$P(j|Op_{mute}, i)$: cette distribution revient, en termes de calculs, à l'emploi de $P(i|Op_{mute}, j)$. Mais en définissant une probabilité sur les sommets d'arrivée des arcs étant donné un sommet parent, nous irions à l'encontre du principe de notre modélisation qui utilise les calculs des scores locaux sur les sommets d'arrivée. La même remarque, portant sur la modélisation employée, est valable pour la distribution $P(Op_{mute}, j|i)$.

Compte tenu des remarques précédentes, nous avons décidé d'implémenter un mécanisme de contrôle sur la distribution de probabilités $P(i|Op_{mute}, j)$ et donc influer sur le choix d'un sommet de départ étant donné le sommet d'arrivée, pour une opération donnée.

Nous pouvons d'ores et déjà déterminer le code de l'algorithme 13, lequel décrit le déroulement d'une phase de mutation pour une structure donnée.

Algorithme 13 Déroulement de la phase de mutation d'un individu

```
1: Pour j = 1 ... n Faire
2:   Si Π_j mute avec une probabilité P_mute Alors
3:     éliciter une opération de mutation Op_mute parmi les opérations réalisables sur Π_j
4:     appliquer Op_mute(i,j) avec la probabilité P(i|Op_mute, j)
5:   Fin Si
6: Fin Pour
```

Détermination de la distribution $P(i|Op_{mute}, j)$

Les probabilités de sélection de l'ensemble Π_j sont uniformes et valent P_{mute} fixée. Le choix de l'opération entreprise est effectué de manière uniforme sur l'ensemble des opérations réalisables sur Π_j.

La seule probabilité nécessitant un calcul et que nous tenons à jour est $P(i|Op_{mute}, j)$.

La distribution étant établie, pour un ensemble Π_j et une opération Op_{mute} donnés, sur l'ensemble des sommets de départ des arcs pour lesquels l'opération Op_{mute} est réalisable, elle doit respecter :

$$\sum_{Op_{mute}} \delta^{(i,j)}_{Op_{mute}} P(i|Op_{mute}, j) = 1$$

avec

$$\delta^{(i,j)}_{Op_{mute}} = \begin{cases} 1 \text{ si l'opération } Op_{mute}(i,j) \text{ est réalisable,} \\ 0 \text{ sinon.} \end{cases}$$

Il est impossible de définir une distribution de probabilité $P(i|Op_{mute}, j)$ fixe, employable à chaque instant par l'ensemble des individus de la population. La diversité des individus impose le calcul de $P(i|Op_{mute}, j)$ pour chaque opération de mutation, pour chaque individu, en fonction des opérations réalisables sur l'ensemble de sommets considéré. Nous proposons de manipuler/définir les différentes probabilités $P(i|Op_{mute}, j)$ à partir de coefficients $\zeta(i, j, Op_{mute}(i, j))$, $1 \leq i, j \leq n$, $i \neq j$ sur lesquels portera le mécanisme de contrôle. La distribution de probabilités se calcule selon l'équation 7.1 :

$$P(i|Op_{mute}, j) = \frac{\zeta(i, j, Op_{mute}(i, j))}{\sum \delta^{(i,j)}_{Op_{mute}} \zeta(i, j, Op_{mute}(i, j))}, \quad \delta^{(i,j)}_{Op_{mute}} = 1 \tag{7.1}$$

Avant de déterminer le processus de calcul des coefficients ζ, et afin qu'aucune opération de mutation ne devienne impossible (probabilité nulle) où ne prédomine l'ensemble des opérations sur un ensemble Π_j (probabilité égale à 1), nous imposons la contrainte suivante sur les coefficients :

$$0,01 \leq \zeta(i, j, Op_{mute}(i, j)) \leq 0,9, \quad \forall 1 \leq i, j \leq n, Op_{mute}(i, j)$$

À l'initialisation, en l'absence de connaissances *a priori*, les $\zeta(i, j, Op_{mute}(i, j))$ sont définis de manière uniforme :

$$\zeta(i, j, Op_{mute}(i, j)) = \frac{1}{n-1} \quad \forall 1 \leq i, j \leq n, Op_{mute}(i, j)$$

Mécanisme de contrôle des coefficients ζ

Maintenant que nous avons établi par quels moyens nous pouvons influencer l'orientation de l'exploration de l'espace, nous devons déterminer comment quantifier cette influence.

Pour cela, il est nécessaire de permettre aux coefficients ζ de prendre en compte aussi bien la qualité des mutations que la fréquence de celles-ci. Au terme de chaque génération, les coefficients ζ sont réévalués :

- seuls les coefficients associés à des opérations de mutation ayant été appliquées au moins une fois sont réévalués ;
- pour un coefficient ζ en cours de réévaluation, la modification de sa valeur est une fonction de la soustraction ω du nombre d'applications infructueuses du nombre d'applications fructueuses de cette opération sur l'arc associé ayant eu lieu au cours de cette génération.

La réévaluation des coefficients ζ est fonction d'un paramètre γ, représentant l'amplitude de la variation du coefficient ζ, et est définie par l'équation 7.2 :

Soit ω=nombre d'applications fructueuses de Op_{mute}-nombre d'applications dommageables de Op_{mute} (durant la génération en cours).

$$\zeta(i, j, Op_{mute}(i, j)) \leftarrow \begin{cases} min(\zeta(i, j, Op_{mute}(i, j)) \times (1 + \gamma)^{\omega}, 0, 9) & \text{si } \omega > 0, \\ max\,(\zeta(i, j, Op_{mute}(i, j)) \times (1 - \gamma)^{\omega}, 0, 01) & \text{sinon.} \end{cases} \tag{7.2}$$

Notons que fixer $\gamma = 0$ revient à employer l'algorithme génétique simple, étant donné que la valeur des coefficients n'est plus modifiée.

Une valeur élevée pour un coefficient ζ – et donc une probabilité $P(i|Op_{mute}, j)$ elle-même élevée – a pour conséquence la propagation du caractère associé chez les individus pour lesquels l'opération est réalisable. Pour un individu sur lequel l'opération considérée ne peut être appliquée – si elle a déjà été appliquée à l'individu au cours d'une génération antérieure, par exemple – la valeur de $\zeta(i, j, Op_{mute}(i, j)$ n'a pas de conséquence sur les probabilités d'application des autres opérations réalisables (puisque l'évaluation de la distribution de probabilités ne tient compte que de ces dernières).

Alors que la population converge vers un optimum (local ou global), la mise à jour régulière des coefficients ζ va tendre à graduellement favoriser ou du moins uniformiser les probabilités $P(i|Op_{mute}, j)$ des différentes opérations possibles et éviter ainsi une convergence prématurée de l'algorithme comme cela peut être observé pour des méthodes où la probabilité de mutation est strictement décroissante [Glickman et Sycara, 2000].

Le comportement attendu de l'algorithme, après mise en place de notre méthode d'orientation des opérations de mutation, est une accélération de la convergence ainsi qu'une tendance moindre à une convergence prématurée, du fait d'un suivi des résultats des différentes mutations au cours de l'évolution.

La section suivante consiste en un rappel des différents points évoqués, à travers une description détaillée de l'implémentation de notre méthode.

Comparaison avec un algorithme EDA

Étant donné que nous définissons une distribution de probabilités que nous modifions au long du fonctionnement d'un algorithme évolutionnaire, il est naturel de se demander quelles sont les différences entre notre approche et une méthode de type EDA telles que celles décrites dans la section 5.4.2.

La distribution $P(i, j, Op_{mute})$ que nous définissons n'est réévaluée que partiellement à chaque génération (les seules probabilités réévaluées sont celles des opérations ayant été effectuées), tandis qu'un algorithme à estimation de densité réévalue l'ensemble des probabilités d'apparition des différents caractères.

De plus, notre méthode définit une distribution de probabilités sur les différentes opérations de mutation et non sur l'apparition de telle ou telle caractéristique d'un individu. Cette différence, bien que conceptuelle, fait que notre algorithme influe sur l'orientation de l'évolution et non sur la population même. Remarquons au passage que la population, dans notre cas, est

maintenue et évoluée d'une génération à l'autre, contrairement à un algorithme EDA où seule la distribution de probabilités est évoluée.

Enfin, la distribution $P(i, j, Op_{mute})$ ne définit pas, à proprement parler, la probabilité d'existence de l'arc $a(i, j) = X_i \rightarrow X_j$.

La probabilité d'existence de l'arc $a(i, j)$ ne tient en effet pas compte des éléments suivants :
- l'arc $a(i, j)$ peut déjà exister dans la population, puisque celle-ci est maintenue et évoluée par l'algorithme ;
- un arc non-existant dans la population peut apparaître suite à une opération d'ajout de $a(i, j)$ mais aussi suite à l'inversion de l'arc $a(i, j)$, si ce dernier existe ;
- l'absence de $a(i, j)$ devrait tenir compte non seulement de ses probabilités d'ajout, de soustraction et d'inversion mais aussi de l'opérateur de réparation qui peut l'effacer s'il s'avère que $a(i, j)$ est l'arc portant le moins d'information mutuelle au sein d'un circuit quelconque.

En résumé, notre méthode consiste avant tout en une stratégie de guidage de l'évolution via une influence exercée sur l'opérateur de mutation et non, comme dans une stratégie de type EDA, en l'évolution de la distribution de probabilités sur les caractéristiques des meilleurs individus jusqu'alors générés.

Implémentation

À l'initialisation de l'algorithme, une probabilité de mutation P_{mute} commune à l'ensemble des individus, pour chacun des ensembles $\Pi_i, i \in 1, \ldots n$, est fixée.

Nous définissons trois matrices, Z_A, Z_S et Z_R.Chacune de ces matrices est affiliée à un parmi trois ensembles de coefficients ζ selon l'opération de mutation considérée :
- Z_A : matrice des coefficients ζ affiliés à l'opération d'ajout d'un arc ;
- Z_S : matrice des coefficients ζ affiliés à l'opération de soustraction d'un arc ;
- Z_R : matrice des coefficients ζ affiliés à l'opération d'inversion d'un arc ;

Ces trois matrices tiennent pour l'ensemble de la population et définissent une pondération pour l'opération de mutation affiliée ainsi que pour les différents arcs sur lesquels cette dernière peut être appliquée.

Le fonctionnement de l'algorithme est le suivant :
- lors de chaque phase de mutation, pour un individu donné, chaque ensemble des différents ensembles de nœuds parents de chaque ensemble de nœuds prédécesseurs (ou parents) $\Pi_j, j \in 1 \ldots n$ de celui-ci subit une opération de mutation avec la probabilité P_{mute} ;
- quand un ensemble Π_j doit muter, un type d'opération Op_{mute} est choisi aléatoirement parmi les types d'opération possibles sur Π_j ;
- une fois l'opération déterminée, l'arc $X_j \leftarrow X_i$ sur lequel va porter celle-ci est choisi aléatoirement, en fonction des coefficients normalisés correspondant à Op_{mute} et au couple (X_i, X_j) ;
- après chaque opération de mutation, nous déterminons si oui ou non cette modification a été fructueuse (*i.e.* a-t-elle permis une amélioration du score *global* de l'individu ?). Un compteur $\omega_{ij}^{Op_{mute}}$ lié au couple de sommets (i,j) et à l'opération de mutation Op_{mute} concernée est incrémenté (opération bénéfique) ou décrémenté (dégradation du score) ;

– lorsque tous les individus ont muté, les différents compteurs rendent compte du nombre de fois où les différentes opérations se sont révélées bénéfiques (ou dommageables) : ces décomptes servent à mettre à jour les coefficients ζ des trois matrices Z_A,Z_S et Z_R.

Une description plus formelle de cette méthode est donnée par les algorithmes 14 et 15. L'algorithme 14 décrit le déroulement de notre méthode durant la phase de mutation, au cours de laquelle sont observées les conséquences des différentes opérations de mutation.
L'algorithme 15 décrit la procédure de mise à jour des coefficients ζ.

Algorithme 14 Phase de mutation

Entrée: Un individu I, trois matrices de coefficients Z_A, Z_S et Z_R, trois matrices de décompte Ω_A, Ω_S et Ω_R correspondant aux opérations d'(A)jout, de (S)oustraction ou d'inversion (*(R)eversal*) d'un arc.

Sortie: Individu muté $I' = \{\mathcal{G}_I, Score(\mathcal{G}_I)\}$, matrices des coefficients ζ et de décompte mises à jour.

$\omega_{Op_{mute}}(i,j) \leftarrow 0, \forall (i,j) \in \{1,\ldots,n\}, \forall Op_{mute} \in \{A,S,R\}$
Pour $j = 1 \ldots n$ **Faire**
  Si Π_j mute **Alors**
    Choisir une opération de mutation $Op_{mute}, Op_{mute} \in \{A,S,R\}$ parmi celles possibles
    Effectuer l'opération $Op_{mute}(i,j)$ sur l'arc (X_i, X_j) avec la probabilité :

$$p = \frac{\zeta(i,j,Op_{mute}(i,j))}{\sum \delta^{(i,j)}_{Op_{mute}} \zeta(i,j,Op_{mute}(i,j))}$$

    $I \leftarrow I'$, individu modifié par Op_{mute}
    Si $Score(\mathcal{G}_{I'}) > Score(\mathcal{G}_I)$ et $\mathcal{G}_{I'}$ sans circuit **Alors**
      $\omega_{ij}^{Op_{mute}}(i,j) \leftarrow \omega_{ij}^{Op_{mute}} + 1$
    Sinon
      Si $Score(I') < Score(I)$ **Alors**
        $\omega_{ij}^{Op_{mute}} \leftarrow \omega_{ij}^{Op_{mute}} - 1$
      Fin Si
    Fin Si
  Fin Si
Fin Pour

7.3 Expérimentation

Des tests préliminaires, présentés en annexes C, ont penché en faveur d'une valeur de 0,5 pour γ. Cette valeur a été employée pour les tests et comparatifs présentés dans le chapitre 8.

Algorithme 15 Mise à jour des matrices de coefficients

Entrée: Matrices des coefficients $\zeta : Z_A$, Z_S et Z_R, matrices de décomptage $\{\Omega_A, \Omega_S, \Omega_R\}$ issues des mutations, paramètre γ.

Sortie: Matrices de coefficients Z_A,Z_S et Z_R, mises à jour.

Pour $i = 1 : n$ **Faire**
  Pour $j = 1 : n$ **Faire**
    Pour $Op_{mute} \in \{A, S, R\}$ **Faire**
      Si $\omega_{ij}^{Op_{mute}}(i, j) > 0$ **Alors**
        $\zeta(i, j, Op_{mute}(i, j)) \leftarrow min(\zeta(i, j, Op_{mute}(i, j)) * (1 + \gamma)^{\omega_{ij}^{Op_{mute}}(i,j)}, 0.9)$
      Sinon
        Si $\omega_{ij}^{Op_{mute}}(i, j) < 0$ **Alors**
          $\zeta(i, j, Op_{mute}(i, j)) \leftarrow max(\zeta(i, j, Op_{mute}(i, j)) * (1 - \gamma)^{\omega_{ij}^{Op_{mute}}(i,j)}, 0.1)$
        Fin Si
      Fin Si
    Fin Pour
  Fin Pour
Fin Pour

Chapitre 8

Expérimentations

L'ensemble des méthodes décrites dans nos travaux ont été implémentées et testées à partir de deux toolbox Matlab spécialement dédiées aux réseaux bayésiens. La première est la *Bayesian Net Toolbox* de K. P. Murphy [Murphy, 2001]. La deuxième est la *Structure Learning Package* (SLP) de P. Leray et O. François [Francois et Leray, 2004], conçue en complément de la BNT et implémentant les principaux algorithmes d'apprentissage de structure.

Ce chapitre est consacré à la présentation des résultats de nos différentes méthodes ainsi qu'aux résultats renvoyés par certains des principaux algorithmes d'apprentissage de structure.

8.1 Objectifs et méthodes

Bien que nous ayons déjà documenté la problématique de l'apprentissage de structures de réseaux bayésiens dans les premiers chapitres de ce travail de thèse, nous en rappelons le déroulement habituel.

L'apprentissage de la structure d'un réseau bayésien, tel qu'abordé dans ce chapitre, revient à :

- prendre en entrée une base d'apprentissage constituée d'exemples d'instanciation jointe des variables du domaine modélisé
- déterminer les relations conditionnelles entre les variables du modèle considéré,
  - soit à partir de tests statistiques effectuées sur plusieurs sous-ensembles des variables ;
  - soit à partir de mesures d'adéquation entre une structure candidate et la base d'apprentissage.
- les différentes méthodes, dans nos évaluations, utilisent des paramètres et/ou des méthodes d'initialisation particulières, ces éléments seront précisées au moment opportun ;
- au final, les structures apprises seront comparées afin de déterminer les qualités respectives des différents algorithmes employés, ces comparaisons seront effectuées à partir d'éléments de mesure précisés et justifiés dans ce chapitre ;

Nous précisons la notion de l'apprentissage de structure dans le cadre de nos expérimentations car celle-ci ignore certains éléments documentés ou, du moins, évoqués dans les chapitres de l'état de l'art tels que :

– prise en compte d'un *a priori* sur la structure recherchée, fourni par un expert ;
– emploi, pour l'apprentissage de bases de données incomplètes ;
– détection d'éventuelles variables latentes.

Maintenant que nous avons établi la démarche suivie lors de l'apprentissage, nous présentons dans la section suivante les méthodes employées ainsi que les protocoles associés.

8.1.1 Méthodes d'apprentissage employées

Dans un souci de comparaison avec les méthodes existantes, nous avons employé, conjointement aux différentes méthodes évolutionnaires que nous avons mises au point, quelques unes des méthodes d'apprentissage de structures, parmi les plus usitées :

– l'algorithme K2 ;
– l'algorithme glouton sur l'espace des structures, noté GS ;
– l'algorithme glouton sur l'espace des graphes équivalents, noté GES ;
– l'algorithme MWST ;
– l'algorithme PC.

On peut remarquer que les méthodes avec lesquelles nous nous comparons sont, exception faite de l'algorithme PC, des méthodes de la famille *search and score* parcourant l'espace des structures candidates à l'aide d'une mesure d'évaluation.

Les méthodes basées sur l'emploi d'un score présentées dans ce chapitre sont toutes utilisées en association avec le critère BIC, dont les caractéristiques sont rappelées plus loin dans ce chapitre. Il est bien entendu possible d'utiliser la plupart de ces méthodes avec d'autres scores ou critères, mais nous souhaitons ici avant tout comparer le comportement des algorithmes lors du parcours de l'espace de recherche et nous utilisons donc le même critère pour chacun d'eux.

Une remarque doit cependant être émise à l'égard de l'algorithme *PC*. Cet algorithme (cf. section 4.2.1) mesure l'indépendance de deux variables conditionnellement aux différents sous-ensembles de variables du domaine. Même en limitant le nombre de nœuds prédécesseurs potentiels pour la structure recherchée, la recherche de structures de grandes tailles (telles que les structures INSURANCE et ALARM) s'effectuait dans des délais rendant l'évaluation de l'algorithme inintéressante, comparativement aux autres méthodes (nous avons préféré arrêter les tests avant leur complétion, celle-ci étant supérieure à 24h là où certaines méthodes telles que MWST ne nécessitaient que quelques secondes). Nous n'avons employé cette méthode que lors de l'apprentissage de la structure du réseau ASIA.

Les méthodes que nous venons d'énumérer sont, de même, comparées à nos quatres algorithmes évolutionnaires d'apprentissage :
– l'algorithme génétique simple, que nous désignerons par la suite par *AG* ;
– l'algorithme génétique panmictique combiné à une stratégie de *niching* séquentiel, noté AG_{penal} ;
– l'algorithme génétique avec adaptativité de la mutation, noté AG_{memo} ;
– l'algorithme génétique combinant le mécanisme de *niching* séquentiel avec un schéma de distribution de la population en îlots que nous notons AG_{dist}.

L'ensemble de ces méthodes sont comparées sur le plan de la qualité des structures apprises, structures correspondant aux modèles décrits ci-après.

8.1.2 Les réseaux appris

Nous appliquons les différents algorithmes à la recherche de structures de complexités croissantes. Ces structures sont celles des réseaux suivants :

ASIA : [Lauritzen et Speigelhalter, 1988] composé de 8 variables et de 8 arcs, voir figure 8.1 ;

Insurance : [Binder et al., 1997] composé de 27 variables et de 52 arcs, voir figure 8.2 ;

ALARM : [Beinlich et al., 1989] composé de 37 variables et de 46 arcs, voir figure 8.3.

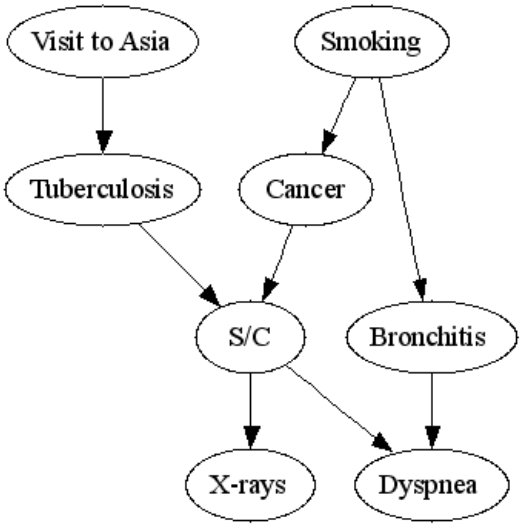


Figure 8.1 – Structure du réseau ASIA.

Le réseau ASIA est un exemple simple de structure couramment employé dans la littérature. Il est caractérisé par une liaison conditionnelle que l'on peut qualifier de très faible entre les variables *Visit to Asia* et *Tuberculosis* (les probabilités d'être atteint de tuberculose selon que l'on ai été ou non en Asie sont, respectivement, de 5% et de 1%) et de deux V-structures entre les variables *Tuberculosis*, *Tuberculosis or Cancer* et *Cancer*, d'une part, et entre les variables *Tuberculosis or Cancer*, *Dyspnea* et *Bronchitis*, d'autre part.

Le réseau INSURANCE constitue un cas d'étude intéressant ; les liaisons au sein du modèle sont nombreuses, comparativement au nombre de variables et sont difficiles à détecter du fait de la probabilité faible de survenance de certaines instanciations (la survenance d'un vol, représenté par la variable *Theft*, est très faible, en général).

Le réseau ALARM, enfin, constitue un compromis entre les deux réseaux précédents : il s'agit d'un réseau de grande taille (aux relations conditionnelles toutefois plus facilement décelables que pour le réseau INSURANCE) et, tout comme ASIA, amplement utilisé dans la littérature.

Outre leurs caractéristiques propres, ces modèles ont servi à de nombreuses reprises pour l'évaluation de méthodes d'apprentissage de structures. Bien qu'une comparaison directe ne soit pas applicable, du fait de l'emploi de bases d'apprentissage particulières selon les travaux,

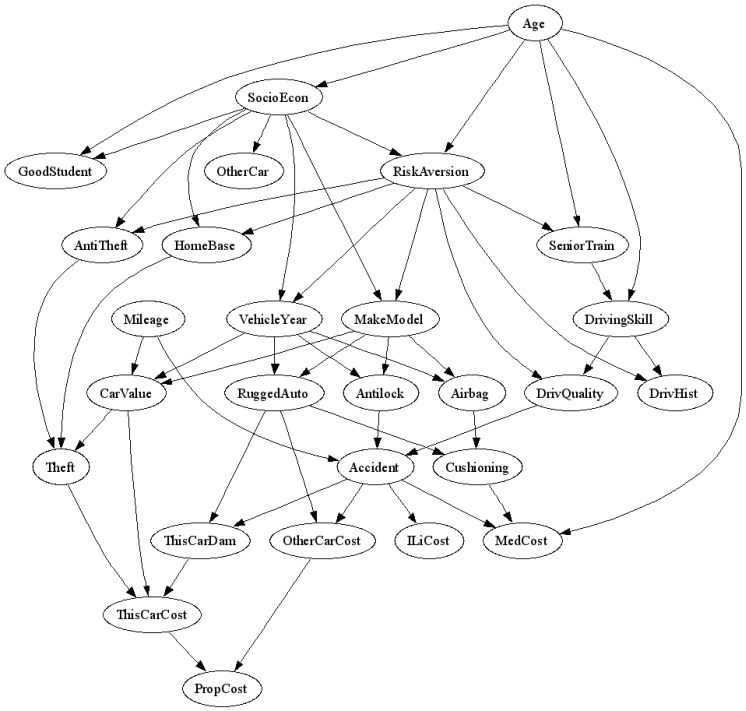


Figure 8.2 – Structure du réseau Insurance.

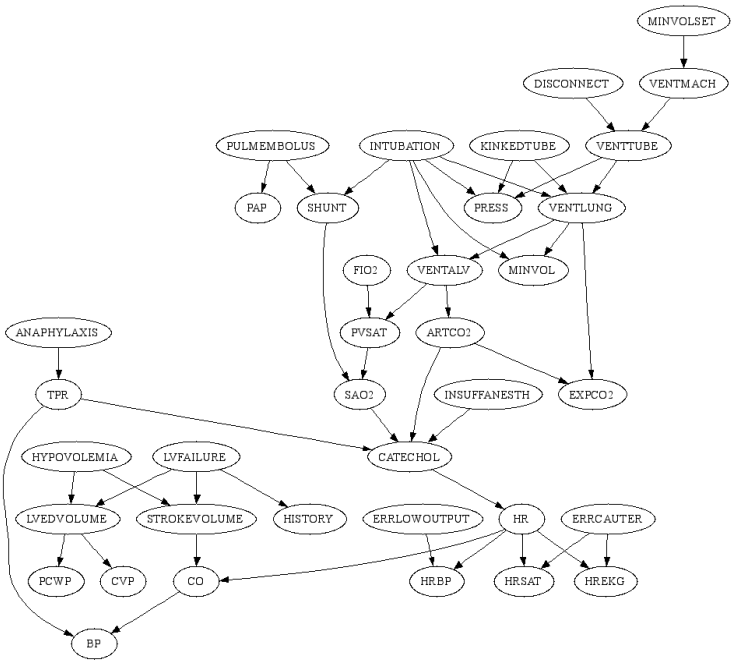


Figure 8.3 – Structure du réseau ALARM.

l'utilisation de ces modèles permet de se figurer la qualité de nos méthodes par rapport aux méthodes passées (et à venir, sans doute).

Nous utilisons chacun de ces réseaux afin de synthétiser :
- quatre ensembles de bases d'apprentissage pour chaque réseau. Chaque ensemble contenant un certain nombre de bases, contenant le même nombre de cas (250, 500, 1000 ou 2000 cas) ;
- une base unique, de grande taille (20 000 ou 30 000 cas) pour chaque réseau. Cette base a pour objectif, de par sa taille, d'être suffisamment représentative des dépendances conditionnelles du réseau dont elle est originaire.

L'ensemble de ces bases est obtenu par échantillonnage probabiliste logique (ou *probabilistic logic sampling*) [Henrion, 1988] : la valeur des sommets n'ayant pas de prédécesseurs est fixée aléatoirement, en accord avec les tables de probabilités du réseau d'origine, puis les variables restantes sont échantillonnées suivant le même principe, en tenant compte des valeurs des sommets prédécesseurs.

Nous utilisons plusieurs bases d'apprentissage pour un réseau et un nombre de cas par base donnés, afin de réduire tout biais consécutif à une erreur d'échantillonnage. En effet, dans le cas de bases de taille limitée, il se peut (et il est même fréquent) que les statistiques pouvant être extraites de ces bases ne représente pas exactement les dépendances conditionnelles présentes dans le réseau d'origine. Par exemple, si l'on considère le réseau ASIA, une base ne contenant que 100 cas échantillonnés depuis le réseau d'origine, peut ne comporter aucune instanciation du domaine pour laquelle la variable "TUBERCULOSE" est vraie.

Après apprentissage auprès des bases de taille limitée, les scores BIC des structures retournées par les différentes méthodes sont obtenus à partir de la base de grande taille évoquée précédemment, afin d'estimer des mesures qualitatives vis-à-vis de la distribution de probabilités qu'elle représente (proche, du fait du nombre élevé de cas, de celle du modèle d'origine).

8.1.3 Mesures utilisées

Afin de mesurer les performances des algorithmes évolutionnaires, nous utilisons différentes valeurs (nombre d'itérations à l'optimal, convergence du score BIC en fonction des itérations) ne pouvant s'appliquer aux autres méthodes, non stochastiques. Néanmoins, afin de pouvoir comparer notre travail à ces algorithmes, nous disposons d'autres critères.

Les problèmes inhérent à une mesure de distance entre deux structures ont été abordés dans le chapitre 6. La conséquence, dans la littérature consacrée à l'apprentissage de structures, notamment, est que plusieurs mesures doivent fréquemment être employées afin de permettre au lecteur d'en faire une synthèse et d'interpréter la qualité des résultats.

Score BIC

Présenté dans la section 4.3, le critère BIC est un critère d'information permettant de mesurer, ici, l'adéquation entre un réseau bayésien et une base de cas. Ce critère applique le principe de

parcimonie en incluant dans son calcul une pénalité, fonction de la complexité structurelle du modèle.

Le critère BIC se présente numériquement sous la forme d'un score négatif, que nous cherchons à maximiser.

Le critère BIC présente l'avantage d'être décomposable, équivalent et consistant (cf. section 4.3). La propriété de consistance revenant, pour un score, à délivrer le score maximal au modèle effectivement sous-jacent à la base d'information à notre disposition. Cette affirmation devant être néanmoins tempérée dans le cas de bases de données de taille limitée. Dans ce cas, il se peut fort bien qu'une structure différente de la structure d'origine parvienne à mieux modéliser les indépendances présentes et, donc, obtenir un meilleur score que cette dernière. Pour cette raison, les valeurs des scores BIC données dans ce chapitre le sont par rapport à des bases de données de grande taille, indépendamment des bases ayant servi à l'apprentissage (excepté, nous le verrons, pour les mesures de divergence entre les distributions représentées où la base ayant servi à apprendre la structure est aussi employée pour apprendre les paramètres du réseau construit). Ceci permet de se représenter une notion effective de la similarité entre le modèle appris et le modèle réel, lequel a de très fortes chances d'obtenir le meilleur score dans ces conditions.

Distance graphique à la structure d'origine

Du fait de l'emploi de bases de taille limitée lors de l'apprentissage, nous pouvons fort bien obtenir des modèles proches graphiquement de la structure d'origine mais obtenant néanmoins des scores BIC inférieurs en raison du jeu complexe des indépendances conditionnelles représentées.

Pour compléter notre ensemble de mesures nous avons calculé la distance d'édition entre deux GOSC $\mathcal{G}_1$ et $\mathcal{G}_2$ en fonction de quatres termes D, $\oplus$, Inv et $\ominus$, où :
- D : Nombre total d'arcs différents entre $\mathcal{G}_1$ et $\mathcal{G}_2$;
- $\oplus$: nombre d'arcs présents dans $\mathcal{G}_1$ et absents dans $\mathcal{G}_2$;
- Inv : nombre d'arcs inversés dans $\mathcal{G}_1$ par rapport à $\mathcal{G}_2$;
- $\ominus$: nombre d'arcs absents de $\mathcal{G}_1$ et présents dans $\mathcal{G}_2$.

Les distances affichées dans les différentes tables sont égales, pour notre algorithme, à la moyenne de ces distances sur les différentes instances effectuées pour une taille de base d'apprentissage et pour un réseau donné.

Ces distances, conjuguées aux informations apportées par le score BIC, permettent de mieux se représenter la qualité des structures renvoyées par les différentes méthodes.

À noter, cependant, que le détail des différences en fonction de leur qualité (ajout, inversion ou absence d'arc) n'est pas superflu ; nous avons précisé, dans la section 4.4.4, la notion de réseaux équivalents. Dans une même classe d'équivalence – ensemble de *GOSC* représentant le même jeu d'indépendances conditionnelles, deux graphes peuvent présenter des arcs inversés les uns par rapport aux autres tout en conférant la même représentation des indépendances conditionnelles au sein du domaine. Sur le graphe partiellement orienté représentant la classe d'équivalence des deux graphes, ceci se traduit par un arc non orienté. La conséquence est que

les deux graphes obtiennent le même score BIC (score équivalent) mais présentent une distance graphique non nulle compte tenu des inversions d'arcs entre les deux *GOSC* (l'inversion).

En utilisant le critère de distance structurelle, il est donc important de conserver à l'esprit que les inversions peuvent aussi bien être sans conséquences (l'inversion correspond à un arc non orienté dans le graphe équivalent) que plus grave (l'inversion crée ou détruit une V-structure).

Divergence de Jensen-Shannon

La divergence de Jensen-Shannon, apparentée à la divergence de Kullback-Leibler – toutes deux présentées en annexe A –, permet de mesurer la dissimilarité entre deux distributions de probabilités. Adaptée pour deux réseaux bayésiens $\mathcal{B}_1$ et $\mathcal{B}_2$, elle s'écrit :

$$\begin{aligned} JS(\mathcal{B}_1 \| \mathcal{B}_2) &= \frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j^1=1}^{q_i^1}\sum_{k=1}^{r_i}\theta_{ijk}^1 log\left(\frac{2\times\theta_{ijk}^1}{P(X_i=k|\Pi^1(X_i)=j^1,\mathcal{G}_2,\Theta_2)+\theta_{ijk}^1}\right)\right) \\ &+ \left(\sum_{i=1}^{n}\sum_{j^2=1}^{q_i^2}\sum_{k=1}^{r_i}\theta_{ijk}^2 log\left(\frac{2\times\theta_{ijk}^2}{P(X_i=k|\Pi^2(X_i)=j^2,\mathcal{G}_1,\Theta_1)+\theta_{ijk}^2}\right)\right) \end{aligned}$$

où $\Pi^k(X_i)$ désigne l'ensemble des parents du nœud X_i dans la structure $\mathcal{G}_k$ du réseau $\mathcal{B}_k$.

Cette divergence permet de mesurer la représentativité de la loi encodée par un réseau bayésien appris avec celle du réseau original. En partant d'une structure apprise et d'une base représentative de la distribution de probabilités d'origine, nous créons un réseau bayésien $\mathcal{B}_2$ dont nous calculons la divergence avec le réseau d'origine.

Le calcul de la divergence de Jensen-Shannon est cependant très long. Il est en effet nécessaire de calculer les probabilités de survenance de l'ensemble des instances du domaine répertoriée dans la base de données, pour chaque modèle.

Nous n'avons employé la divergence de Jensen-Shannon que pour comparer les résultats obtenus sur l'apprentissage du réseau ASIA, les calculs requis pour le calcul de cette divergence dans le cas de structures plus complexe s'étant avérés trop longs pour être efficacement effectués sur l'ensemble des tests réalisés.

Conjointement à la distance d'édition graphique, la divergence de Jensen-Shannon permet de décomposer le résultat du score BIC lequel tient compte à la fois de la représentativité du modèle et de sa complexité graphique.

Mesures statistiques

Nous verrons dans la section consacrée à l'analyse des résultats que les différences de représentativités des bases d'apprentissage que nous employons a pour conséquence une dispersion assez importante des scores des solutions d'une méthode donnée lorsque celles-ci ont été apprises à partir d'une base de taille limitée. Afin de pouvoir faire ressortir les différences décelables entre les différents résultats, nous emploierons, au sein des tables de résultats, un

test statistique non paramétré permettant de pouvoir affirmer l'existence d'une différence significative entre deux groupes de scores obtenus. Ce test, le test de Mann-Whitney (cf. annexe A), est ici employé sous l'hypothèse bilatérale de l'existence d'une différence significative entre les deux séries de scores comparés, avec un seuil de confiance de 5%.

8.1.4 Protocoles expérimentaux

Le paramétrage des algorithmes, sauf précision contraire, a été le suivant lors des tests :

8.1.4.1 Protocole - méthodes usuelles

K2 : Cet algorithme nécessite l'entrée d'un ordre topologique sur les sommets du graphe recherché. Nous avons utilisé dans ce but deux types d'initialisation :
– l'ordre topologique d'un arbre retourné par la méthode MWST (méthode K2-T) ;
– un ordre topologique aléatoire (méthode K2-R).
Pour chaque instance de K2-R – i.e. pour chaque base d'apprentissage considérée –, nous procédons à $5 \times n$ initialisations aléatoires pour ne retenir que celle renvoyant le meilleur score BIC ;

GS : l'algorithme glouton, noté GS, est initialisé avec un arbre retourné par la méthode MWST, dont la racine est aléatoirement fixée ;

GES : l'algorithme glouton sur l'espace des graphes équivalents, GES, est initialisé avec la structure vide ;

MWST : l'algorithme MWST est initialisé avec un nœuds racine sélectionné aléatoirement (ceci n'a pas d'influence sur le score de la structure obtenue) ;

PC : les indépendances conditionnelles sont déterminées par l'intermédiaire du test du Chi^2 de Pearson (cf. annexes A.2.0.1). Le seuil de confiance α associé est fixé à 0,05. Le nombre maximal de sommets prédécesseurs, pour une variable de la structure recherchée est, quant à lui, fixé à 3.

8.1.4.2 Protocoles - méthodes évolutionnaires

– taille de la population : 150 individus pour les algorithmes ne faisant évoluer qu'une seule population. L'algorithme à population distribuée AG_{dist} fait évoluer, quant à lui, 30 sous-populations réparties de 30 individus chacune ;
– probabilité de mutation : $\frac{1}{n}$;
– probabilité de croisement : 0,80 ;
– politique élitiste : le meilleur individu de la population en cours est conservé à la génération suivante ;
– critère d'arrêt : le nombre d'itérations, fixé à 1000. Excepté pour l'algorithme distribué pour lequel cette limite a été fixée à 300 ;
– initialisation : les populations des différentes méthodes évolutionnaires sont initialisées de la même manière : par l'arbre non-orienté renvoyé par l'algorithme MWST, orienté à partir d'un sommet racine choisi aléatoirement. Nous nous assurons néanmoins que chaque sommet soit choisi au moins une fois en tant que racine.

Le choix des valeurs de paramètres tels que la probabilité de croisement ou celle de mutation proviennent d'une part de certaines conventions dans le domaine des algorithmes évolutionnaires [Bäck, 1993] mais aussi et surtout d'observations empiriques faites au long de nos travaux. Si les résultats de ces (nombreuses) expériences ne sont pas détaillés ici, les résultats de certaines expériences menées afin de fixer une valeur adéquate à certains paramètres spécifiques à nos méthodes – tels que le nombre et la taille des sous populations au sein de l'algorithme AG_{dist} – figurent dans l'annexe C.

Le nombre d'itérations fixé en tant que critère d'arrêt des différents algorithmes a été élicité suivant deux critères :

- une fois une taille de population assurant une bonne convergence de celle-ci, il convient de fixer un nombre maximal d'itérations afin de limiter le nombre total d'évaluations. L'intérêt d'une stratégie évolutionnaire n'étant pas tant de converger mais aussi de pouvoir le faire dans une limite de temps/d'itérations raisonnable ;
- il est cependant intéressant d'assurer un temps assez long à l'évolution afin de pouvoir observer, au final, le *véritable* temps mis par la population à trouver la meilleure solution.

Le choix du nombre d'itérations se veut donc à la fois suffisant pour pouvoir observer et interpréter les performances de la méthode considérée tout en évitant un nombre d'évaluations faussant la comparaison de résultats avec des méthodes gloutonnes.

Outre les paramètres communs aux différentes méthodes évolutionnaires listés précédemment, les algorithmes AG_{penal}, AG_{memo} et AG_{dist} nécessitent eux-mêmes la définition de certains paramètres particuliers. Ces paramètres ont fait l'objet de tests particuliers, eux aussi présentés dans l'annexe C

- pour AG_{penal}, le paramètre Ite_{opt} est fixé à 20 ;
- pour AG_{memo}, le paramètre γ est fixé à 0,5. De plus, nous employons avec cet algorithme une probabilité de mutation de $\frac{1}{n}$ afin de tirer profit du mécanisme d'adaptivité de l'opérateur de mutation ;
- pour AG_{dist} l'intervalle migratoire I_{mig} est réglé à 20 itérations tandis que le taux de migration T_{mig} est de 10%.

Pour chaque réseau appris et pour une taille de base d'apprentissage donné, chaque méthode est exécutée une fois, les résultats sont ensuite moyennés sur l'ensemble des bases employées.

8.2 Apprentissage de la structure ASIA

La structure ASIA étant simple, comparativement aux autres algorithmes figurant dans nos expérimentations, nous avons choisi de ne pas employer les mêmes paramétrages généraux qu'avec les structures complexes comme Insurance. Les paramètres particuliers employés pour ASIA sont les suivants :

- nombre d'itérations : 100 itérations pour AG et AG_{penal}, 50 pour AG_{memo} et 40 pour l'algorithme distribué AG_{dist} ;
- paramètre de mémorisation Ite_{opt} pour l'algorithme AG_{penal} fixé à 10 ;

– nombre de populations en parallèle fixé à 10 pour AG_{dist}.

Nous disposons, pour l'apprentissage, de 30 bases de cas échantillonnées depuis le réseau d'origine et ce, pour chacune des quatre tailles de base considérées.

Résultats et commentaires

	ASIA			
	250	**500**	**1000**	**2000**
AG	−68912 ± 910	−68345 ± 212	−68273 ± 68	−68244 ± 11
AG_{penal}	−68959 ± 919	**-68338** ± 213	−68272 ± 69	**-68241** ± 1
AG_{memo}	−68908 ± 840	−68401 ± 349	**-68274** ± 69	−68243 ± 4
AG_{dist}	**-68857** ± 826	−68340 ± 213	−68273 ± 69	−68242 ± 1
GS	−69197 ± 916	−68514 ± 512†	−68307 ± 95†	−68262 ± 56†
GES	−68907 ± 768	−68422 ± 266	−68291 ± 93	−68251 ± 0
K2-T	−69093 ± 925†	−68447 ± 419†	−68276 ± 68†	−68255 ± 24†
K2-R	−69358 ± 875†	−68617 ± 448†	−68327 ± 129†	−68266 ± 55†
MWST	−70178 ± 546†	−69959 ± 226†	−69931 ± 168†	−69857 ± 60†
PC	−73916 ± 2371†	−72039 ± 1523†	−72592 ± 1643†	−73106 ± 1492†
Original	−68241			
$\mathcal{G}_0$	−88564			

Tableau 8.1 – Moyennes et écart-types arrondis des scores BIC obtenus par les différentes méthodes, pour l'apprentissage de la structure du réseau ASIA à partir de 30 bases de cas distinctes. Les valeurs moyennes minimales, pour chaque taille de base d'apprentissage, sont grisées. À titre indicatif, les scores des structures d'origine et de la structure vie, $\mathcal{G}_0$, sont indiqués en bas de table. Le signe † dans une case signifie que les résultats de la méthode correspondantes sont significativement différents de ceux de la méthode présentant la meilleure moyenne (vérification par un test de Mann-Whitney).

La première table de résultats à notre disposition, la table 8.1, recense les scores des structures obtenues par les différentes méthodes, moyennés (correspondant aux 30 structures obtenues sur chaque base des différents ensembles, pour une taille donnée).

Pour des bases de données de taille faible et peu représentatives, les méthodes évolutionnaires, ainsi que les méthodes gloutonnes GS et GES obtiennent des résultats semblables. Il est à noter que les méthodes GS et GES, nous le verrons dans la section 8.6.2, obtiennent ici leurs résultats en des temps brefs. La structure du réseau ASIA est en effet très simple.

Les méthodes de type K2 (R ou T) sont très rapides, même en tenant compte, pour l'algorithme K2-T de l'exécution préalable de l'algorithme MWST afin d'obtenir un ordre topologique en entrée.

Cependant, lorsque la taille des bases d'apprentissage augmente (et donc, que ces bases deviennent plus représentatives de la distribution de probabilités du modèle recherché), les performances de l'algorithme GS se dégradent comparativement à celles des algorithmes évo-

	ASIA							
	250				**500**			
	D	$\oplus$	*Inv*	$\ominus$	**D**	$\oplus$	*Inv*	$\ominus$
AG	4	0,7	1,3	2	3,1	0,5	1,2	1,3
AG_{penal}	4	0,7	1,3	2	2,8	0,3	1,2	1,3
AG_{memo}	4	0,7	1,3	2	3,5	0,6	1,5	1,3
AG_{dist}	**3,8**	0,7	1,2	1,9	**2,7**	0,3	1,1	1,3
GS	5,1	1	2,1	2	5	1,2	2,2	1,6
GES	3,8	0,6	1,3	1,9	3	0,3	1,3	1,3
K2-T	6,7	1,3	3,3	2	7,2	1,7	4	1,5
K2-R	4,2	0,8	1,4	2	4	0,8	1,8	1,4
MWST	6,1	1	3	2,1	6,7	1	3,8	1,9
PC	7,5	0,1	3	4,4	7,7	0	3,8	3,8
	1000				**2000**			
	D	$\oplus$	*Inv*	$\ominus$	**D**	$\oplus$	*Inv*	$\ominus$
AG	**2,4**	0,2	1	1,2	2,2	0,2	1,1	0,9
AG_{penal}	**2,4**	0,2	1	1,2	**2**	0	1,1	0,9
AG_{memo}	2,9	0,4	1,3	1,2	2,6	0,3	1,3	1
AG_{dist}	2,5	0,2	1,1	1,2	2,2	0,1	1,1	0,9
GS	5	1	2,8	1,2	5,5	1,2	3,3	1
GES	2,4	0,2	1	1,2	**2**	0	1,1	0,9
K2-T	7	1,8	4	2	7,5	1,9	4,6	1
K2-R	3,2	0,7	1,4	1,1	3,4	0,7	1,7	1
MWST	6,1	0,7	3,7	1,7	6,4	0,6	4,2	1,6
PC	6,6	0,1	2,9	3,6	5,4	0	2,4	3

Tableau 8.2 – Différences structurelles moyennes entre les réponses des différents algorithmes et la structure du réseau ASIA à partir de 30 bases de cas distinctes. Les valeurs moyennes minimales sont grisées.

	ASIA			
	250	**500**	**1000**	**2000**
AG	124 ± 50	46 ± 17	23 ± 9	10 ± 3
AG_{penal}	124 ± 49	46 ± 17	23 ± 9	9 ± 3
AG_{memo}	123 ± 49	48 ± 19	23 ± 9	9 ± 3
AG_{dist}	123 ± 49	46 ± 17	22 ± 8	9 ± 3
GS	139 ± 49	54 ± 25	27 ± 11	12 ± 7
GES	126 ± 51	53 ± 24	25±13	10 ± 6
K2-T	149 ± 46	53 ± 22	24 ± 9	10 ± 4
K2-R	134 ± 55	64 ± 21	29 ± 12	13 ± 6
MWST	214 ± 37	174 ± 24	152 ± 14	141 ± 68
PC	392 ± 140	279 ± 82	292 ± 100	332 ± 105

Tableau 8.3 – Moyennes et écart-types arrondis des divergences de Jensen-Shannon des solutions obtenues par les différentes méthodes, pour l'apprentissage de la structure du réseau ASIA à partir de 30 bases de cas distinctes. Pour faciliter leur lecture, les valeurs sont ici arrondies et multipliées par 10^4. Les moyennes minimales, pour chaque taille de base d'apprentissage, sont grisées.

lutionnaires et de l'algorithme GES ; les tests de Mann-Whitney démontrent qu'il existe une différence significative entre les résultats de GS et ceux des autres méthodes.

Les performances de l'algorithme PC sont décevantes sur le plan des scores BIC des structures obtenues. Ce comportement était cependant prévisible, cet algorithme étant le seul à ne pas employer le score BIC en tant que critère de sélection pour les solutions qu'il renvoie.

Les valeurs des écarts types sont relativement élevées, pour l'ensemble des méthodes, lorsque les bases sont de tailles faibles. La représentativité des bases est en effet, dans ce cas, très variable. De manière intuitive, on peut fort bien se représenter le fait qu'une même base de très petite taille a des chances de pouvoir être issue de l'échantillonnage de modèles différents. Ainsi, pour une base d'apprentissage donnée, le modèle correspondant à la meilleure évaluation (modèle représentant le plus simplement et au mieux la probabilité sous-jacente) peut être très différent du modèle à l'origine de la base. Ce comportement se vérifie entre autres par le fait que l'écart type des scores des solutions retournées par une méthode donnée diminue à mesure que la base d'apprentissage employée devient plus grande. Cependant, on peut remarquer que les résultats des méthodes K2-R et K2-T voient aussi leurs écart-types se réduire alors que leur sensibilité à l'ordre topologique qui leur est fourni en entrée devrait, *a priori*, les amener à la réalisation de structures très différentes. En fait, dans le cas d'étude qui nous intéresse, nous procédons à 5*n lancements à partir d'ordres topologiques aléatoires ; dans le cas d'un réseau de petite taille tel qu'ASIA, cela suffit à obtenir, en moyenne, de bons résultats. Nous verrons dans les expériences suivantes que dans le cas de réseaux plus compliqués, ceci peut cependant s'avérer très dommageable pour la qualité des solutions.

Les solutions retournées par les méthodes évolutionnaires dans leur ensemble, pour des bases d'apprentissage de 2000 cas, obtiennent des scores très proches de celui du réseau original, score présenté en bas de la table 8.1, et ce malgré une distance d'édition graphique non nulle. Ceci est expliqué par, en général, l'obtention d'une structure où le lien entre les variables *Visit to Asia* et *Tuberculosis* est absent (de poids faible en terme de probabilité, son ajout dégrade le score BIC en deçà d'une certaine taille pour la base d'apprentissage, en compliquant la structure. Les inversions d'arcs, sont alors à mettre au compte d'arc n'appartenant pas à une V-structure.

La table 8.2, répertoriant les valeurs moyennes des distances d'édition entre les graphes retournés et la solution recherchée, reflètent les résultats qualitatifs de la précédente table en donnant l'avantage aux mêmes méthodes. On peut cependant remarquer que dans le cas de bases de petites tailles (250 ou 500 cas), alors que les scores BIC des structures retournées par l'algorithme glouton ne présentaient pas de différence significative avec ceux des méthodes évolutionnaires ou GES, les structures correspondantes présentent une distance d'édition supérieure, en particulier un nombre d'arcs superflus légèrement supérieur.

Les distances d'édition les plus défavorables sont attribuées aux solutions des algorithmes MWST – conséquence naturelle de la limitation de cet algorithme à l'espace des arbres – et à l'algorithme K2 initialisé par l'ordre topologique induit par l'algorithme MWST. Dans ce dernier cas, la majorité des différences réside dans l'inversion d'arcs

Les méthodes évolutionnaires permettent, au final, d'obtenir des structures, pour le réseau ASIA, de bonne qualité et peu différentes, graphiquement, de la structure d'origine.

La dernière table de résultats, la table 8.2, renvoie les valeurs moyennes des divergences de Jensen Shannon entre les modèles construits à partir des structures renvoyées et des bases d'ap-

prentissage correspondantes (servant alors à apprendre les paramètres des modèles) et celle du réseau ASIA. Dans l'ensemble, les modèles les plus proches sont, ici aussi, renvoyés par les méthodes évolutionnaires et l'algorithme GES. On peut remarquer, de même, que les écart types des divergences des solutions renvoyées par les méthodes évolutionnaires sont aussi réduits, comparativement à ceux correspondant aux autres méthodes, y compris GES. L'algorithme GS, ainsi que les méthodes de type K2 renvoient, elles aussi, des résultats de bonne qualité. Les solutions les plus éloignées de la distribution d'origine sont celles renvoyées par l'algorithme PC. En regard de la table 8.2, les résultats de cet algorithme ont pour particularité, comparativement aux solutions d'autres méthodes de présenter un nombre supérieur d'arcs manquants. Cette méthode semble devoir nécessiter un nombre conséquent d'exemples afin de pouvoir assurer la fiabilité des tests d'indépendance conditionnelle à la base de son fonctionnement, un reproche fréquemment adressé à l'égard des méthodes statistiques d'apprentissage des structures.

L'étude des apprentissages effectués autour de la structure du réseau ASIA permet de se figurer une première idée des qualités intrinsèques aux différents algorithmes que nous employons. Les résultats présentés ici ne permettent pas de distinguer un meilleur comportement de la part d'une méthode parmi les méthodes évolutionnaires et les algorithmes gloutons mais nous pouvons d'ores et déjà remarquer que les méthodes les plus rapides (nous renvoyons le lecteur à la section 8.5 pour une description plus complète des temps de calculs des méthodes répertoriées) ont un revers en ce qu'elles confèrent une représentation limitée (l'algorithme MWST est limité à l'espace des arbres) ou bien font preuve d'une grande sensibilité à l'égard de leurs données d'entrée.

Dans la suite, nous allons tenter de dégager plus précisément les qualités des algorithmes en les confrontant à l'apprentissage de structures plus complexes, telles que celle du réseau Insurance.

8.3 Apprentissage de la structure Insurance

Le réseau Insurance, représenté sur la figure 8.2, est bien plus complexe que le réseau ASIA. Notamment, détail non présenté sur la figure, de nombreuses probabilités conditionnelles au sein de ce réseau ont une valeur très faible et sont donc difficiles à établir à partir d'une base de cas restreinte.

L'apprentissage est effectué, pour chaque taille donnée, sur un ensemble de 10 bases de cas échantillonnées depuis le réseau Insurance. Chaque base d'apprentissage fait ici l'objet d'un seul apprentissage, pour chaque méthode.

Résultats et commentaires

Du fait de la multiplicité des variables, les scores sont plus élevés et nous avons choisi de présenter leurs valeurs moyennes et écart-types divisés par dix, afin de simplifier la lecture de la table 8.4. La même démarche a été suivie pour l'élaboration de la table 8.6, dans la section suivante.

	Insurance			
	250	**500**	**1000**	**2000**
AG	−32135 ± 290	−31200 ± 333	−29584 ± 359	−28841 ± 89†
AG_{penal}	−31917 ± 286	−31099 ± 282	−29766 ± 492	**-28681**±156
AG_{memo}	**-31826**±270	−31076 ± 151	−29635 ± 261	−28688 ± 165
AG_{dist}	−31958 ± 246	**-31075**±255	**-29428**±290	−28715 ± 164
GS	−32227 ± 397	−31217 ± 314	−29789 ± 225†	−28865 ± 151†
GES	−33572 ± 247†	−31952 ± 273†	−30448 ± 836†	−29255 ± 634†
K2-T	−32334 ± 489†	−31772 ± 339†	−30322 ± 337†	−29248 ± 163†
K2-R	−33002 ± 489†	−31858 ± 395†	−29866 ± 281†	−29320 ± 245†
MWST	−34045 ± 141†	−33791 ± 519†	−33744 ± 296†	−33717 ± 254†
Original	−28353			
$\mathcal{G}_0$	−45614			

Tableau 8.4 – Moyennes et écart-types, divisés par 10 et arrondis, des scores BIC des solutions obtenues par les différentes méthodes pour l'apprentissage de la structure du réseau Insurance à partir de 10 bases de cas distinctes. Les valeurs moyennes minimales, pour chaque taille de base d'apprentissage, sont grisées. À titre indicatif, les scores des structures d'origine et de la structure vie, $\mathcal{G}_0$, sont indiqués en bas de table. Le signe † dans une case signifie que les résultats de la méthode correspondantes sont significativement différents de ceux de la méthode présentant la meilleure moyenne (vérification par un test de Mann-Whitney).

La lecture de la table 8.4, répertoriant les scores moyens obtenus par les solutions des différents algorithmes, donne un avantage aux méthodes évolutionnaires. S'il est, comme pour l'apprentissage de la structure du réseau ASIA, impossible de départager clairement les performances des différentes méthodes évolutionnaires, on peut cependant remarquer que ces derniers surclassent en général les algorithmes GES et GS, qui, précédemment, étaient leurs principaux concurrents. Seul l'algorithme glouton sur l'espace des *GOSC* réussit à obtenir d'aussi bons résultats, sur les ensembles de bases de taille limitée (250 et 500).

La lecture de la table 8.5 montrent que, de manière plus ou moins sensible, les algorithmes évolutionnaires employant la méthodes de *niching* séquentiel (nommément AG_{penal} et AG_{dist} retournant des structures présentant le moins de différences structurelles vis-à-vis du réseau d'origine. Le réseau Insurance présente de nombreuses dépendances faibles et, en général, les méthodes employant une fonction d'évaluation et en particulier le score BIC retournent des structures très différentes de la structure d'origine (on pourra préférer, dans ce cas, employer un score pénalisant moins les structures complexes tel que le score BDeu [Delaplace et al., 2007a]). Alors que le réseau Insurance comporte 52 arcs, les différences structurelles les plus importantes se situent aux alentours de 40 arcs différents, essentiellement des arcs manquants.

Il est surprenant de constater que l'algorithme GES, dans le cas de l'apprentissage à partir de bases de 1000 cas, renvoie des solutions moins performantes en terme de scores ainsi qu'un écart-type important. La méthode de construction graduelle de l'algorithme peut très bien avoir amené ce dernier à être bloqué en certains optima locaux, lors de sa recherche.

La table 8.5, répertoriant les différences structurelles entre les solutions retournées et la structure du réseau Insurance, amène aussi une explication quant aux difficultés rencontrées par les différentes méthodes : alors que le score moyen des méthodes renvoyées par l'algorithme AG_{penal}

	Insurance							
	250				**500**			
	D	$\oplus$	*Inv*	$\ominus$	**D**	$\oplus$	*Inv*	$\ominus$
AG	39,6	4,4	7,2	28	34	3,1	7,6	23,3
AG_{penal}	**37**	3,5	7,1	26,4	35,1	3,7	7,4	24
AG_{memo}	37,5	4,3	6,6	26,6	33,9	3,2	7,7	23
AG_{dist}	38,1	3,5	7,5	27,1	**33,3**	3	7,3	23
GS	42,1	4,6	9,4	28,1	37,7	4,5	9,4	23,8
GES	39,5	3,7	7,1	28,7	35,1	3	7,1	25
K2-T	42,7	5,1	8,4	29,2	40,8	5,4	8,8	26,6
K2-R	42,4	4,8	7,2	30,4	41,8	6,5	8,8	26,6
MWST	41,7	4	7,7	30	41,3	3,5	8,3	29,5
	1000				**2000**			
	D	$\oplus$	*Inv*	$\ominus$	**D**	$\oplus$	*Inv*	$\ominus$
AG	39,6	4,4	7,2	28	27,8	4,7	8	15,1
AG_{penal}	30,8	3,8	7,4	19,6	**24,4**	3,4	6,7	14,3
AG_{memo}	31,4	4	8	19,4	27	4,3	8,4	14,3
AG_{dist}	**29,3**	3,6	6,5	19,2	26,6	3,6	8,6	14,4
GS	35,9	5,1	10	20,8	31,9	5,2	11,4	15,3
GES	32,4	4,1	8,1	20,2	27,5	4	8,4	15,1
K2-T	38,7	5,9	11	21,8	34,6	7,3	10,9	16,4
K2-R	39,6	8,3	8,3	23	36,1	8,5	8,5	9,1
MWST	37,7	1,7	8,3	27,7	36,3	1,2	7,9	27,2

Tableau 8.5 – Différences structurelles moyennes entre les structures établies par les différents algorithmes à partir de 10 bases de cas distinctes et la structure du réseau Insurance. Les valeurs moyennes minimales sont grisées.

sont proches du score de la structure originelle du réseau Insurance, les différences structurelles entre ces solutions et la structure d'Insurance demeurent proportionnellement importants avec une moyenne de plus de 24 arcs différents (dont plus de la moitié sont manquants). Une grande partie des liens présents dans la structure d'origine ne sauraient donc, *a priori* être trouvés par l'intermédiaire d'une fonction d'évaluation (cela dépend aussi, comme cela a été mentionné précédemment, de la fonction employée) : les ajouts d'arcs s'avérant plus pénalisants que le montant de la vraisemblance qu'ils apportent au score de la structure.

Les résultats de l'algorithme MWST, bien que médiocres comparativement à ceux des autres méthodes, demeurent stables dès que le nombre de cas dans la base d'apprentissage dépasse 500 cas. Cette observation est valable aussi bien pour le score moyen de ses solutions que pour les différences structurelles entre celles-ci et le réseau Insurance.

Au final, l'apprentissage effectué sur une structure complexe fait ressortir une tendance, pour les algorithmes gloutons (GS et GES) à se retrouver bloqués en un optimum local. L'observation des différences structurelles moyennes entre les solutions renvoyées par GS et celles renvoyées par l'algorithme AG_{memo} pour les cas de bases d'apprentissage de 250 individus, principalement des inversions, alors que les scores obtenus par les deux méthodes ne sont pas significativement différents laisse à penser qu'une méthode stochastique telle qu'un de nos algorithmes est plus apte, dans un cas complexe, à déterminer correctement les arcs d'une structure sur le score.

8.4 Apprentissage de la structure ALARM

Si le réseau ALARM comporte plus de variables que le réseau Insurance (37 contre 27), il comporte moins d'arcs (46 contre 52) et, notamment, moins de relations de probabilité faible que ce dernier.

Chaque ensemble de bases d'apprentissage contient 30 bases échantillonnées depuis le réseau ALARM. Chacune des bases d'un ensemble servant à un seul apprentissage pour chaque algorithme évalué.

Résultats et commentaires

Les valeurs indiquées par la table 8.6 permettent de voir que, en ce qui concerne les scores des solutions renvoyées, les méthodes employant la technique de *niching* séquentiel, combiné ou non à une répartition de la population, obtiennent de très bons résultats. La méthode adaptative AG_{memo} obtient bien, en moyenne, les meilleurs résultats par rapport à des bases d'apprentissage de 500 cas malheureusement, la qualité des résultats de cette solution est irrégulière sur les autres tailles de base possibles ; au point que la même méthode renvoie les plus mauvais résultats, au sein des méthodes évolutionnaires, pour des bases de 1000 cas.

L'algorithme glouton GS renvoie des solutions obtenant de bons scores mais demeurent en retrait face aux méthodes GES et AG_{dist}.

L'algorithme GES renvoie quant à lui des solutions dont les scores sont certes, en moyenne, inférieurs à ceux des solutions renvoyées par AG_{dist} mais le niveau de performances de cet algorithme reste régulier, quelle que soit la taille de base d'apprentissage considérée.

	ALARM			
	250	**500**	**1000**	**2000**
AG	−36239 ± 335	−34815 ± 317	−33839 ± 159	−33722 ± 204†
AG_{penal}	**-36094**±297	−34863 ± 346	−33865 ± 203	−33640 ± 196†
AG_{memo}	−36104 ± 316	**-34791**±340	−33942 ± 198†	−33722 ± 204†
AG_{dist}	−36144 ± 326	−34864 ± 337	**-33723**±251	**-33496**±170
GS	−36301 ± 309†	−35049 ± 380†	−33839 ± 109†	−33638 ± 964†
GES	−36124 ± 315	−34834 ± 288	−33801 ± 562†	−33593 ± 692†
K2-T	−36615 ± 308†	−35637 ± 328†	−34427 ± 200†	−34045 ± 818†
K2-R	−37173 ± 435†	−35756 ± 264†	−34579 ± 305†	−34128 ± 173†
MWST	−37531 ± 185†	−37294 ± 737†	−37218 ± 425†	−37207 ± 366†
Original	−33097			
$\mathcal{G}_0$	−63113			

Tableau 8.6 – Moyennes et écart-types, divisés par 10 et arrondis, des scores BIC des solutions obtenues par les différentes méthodes, pour l'apprentissage de la structure du réseau ALARM à partir de 30 bases de cas distinctes. Les valeurs moyennes minimales, pour chaque taille de base d'apprentissage, sont grisées. À titre indicatif, les scores des structures d'origine et de la structure vie, $\mathcal{G}_0$, sont indiqués en bas de table. Le signe † dans une case signifie que les résultats de la méthode correspondantes sont significativement différents de ceux de la méthode présentant la meilleure moyenne (vérification par un test de Mann-Whitney).

Le principal inconvénient de l'algorithme AG_{memo} semble résider dans le nombre d'inversions d'arcs au sein des solutions qu'il propose. Alors qu'elle a été conçue pour permettre une exploration efficace de l'espace des solutions, il semble que la stratégie appliquée par cet algorithme, en particulier dans le cas de problèmes de dimension croissante, n'a tout simplement pas le temps d'être appliquée. En effet, pour être réellement efficace, cette stratégie se propose d'exploiter l'ensemble des opérations de mutations possibles et intéressantes pour les individus de la population. Or, si le nombre de ces opérations devient trop important, l'énumération et l'évaluation de ces différents mouvements possibles deviennent laborieuses. Quand à la réestimation d'opérations précédemment décrétées comme étant dommageables, elle n'a tout simplement pas l'occasion de prendre place.

Enfin, les méthodes "rapides" (nous verrons cependant que les conditions d'initialisation de ces méthodes vient tempérer l'emploi de cet adjectif) telles que MWST et les méthodes de type K2 se comportent d'une manière similaire à ce qui avait été précédemment observé, à savoir des solutions de facture moyenne, qualitativement inférieures à celles des méthodes précédemment citées.

La table 8.7, combinée à la lecture de la table 8.6, permet des observations intéressantes. En effet, nous avons observé que les scores des solutions obtenues par l'algorithme GES étaient en moyenne inférieurs à ceux des solutions obtenues par voies évolutionnaires. Nous pouvons cependant remarquer qu'en termes de distance d'édition, ces mêmes solutions proposées par l'algorithme GES sont les plus proches du graphe d'origine. Cette observation peut nous amener à penser que la philosophie de l'algorithme GES, consistant à construire graduellement la solution recherchée en respectant certaines règles de construction essentiellement locales (cf. section 4.4.4 mais aussi [Chickering, 2002b]) permet à ce dernier d'obtenir une solution non seulement vraisemblable par rapport aux données mais aussi graphiquement proche de l'opti-

	ALARM							
	250				**500**			
	D	$\oplus$	*Inv*	$\ominus$	**D**	$\oplus$	*Inv*	$\ominus$
AG	34,2	4,8	13,9	15,5	25,7	4,5	10,2	11
AG_{penal}	33,1	4,6	13,5	15	25,6	4,2	10,6	10,8
AG_{memo}	33	4,6	13,4	15	26,2	4	11,5	10,7
AG_{dist}	33,6	4,6	13,8	15,2	25,1	3,7	10,7	10,7
GS	33,7	5	12,6	16,1	30,2	5	13,5	11,7
GES	**32,5**	4,5	12,7	15,3	**23,3**	3,8	8	11,5
K2-T	34,5	5,1	13,1	16,3	35,1	7,2	15,2	12,7
K2-R	36,5	6,6	10,2	19,6	35	8,7	11,3	11,5
MWST	38,5	6,9	14,7	16,9	36,5	4,7	17,1	14,7
	1000				**2000**			
	D	$\oplus$	*Inv*	$\ominus$	**D**	$\oplus$	*Inv*	$\ominus$
AG	19,7	3,7	9	6,9	23	5,3	11,8	5,9
AG_{penal}	22	4,5	10,4	7,1	20,1	4,1	10,2	5,8
AG_{memo}	27	6,4	13,1	7,4	29	7,4	16	6,3
AG_{dist}	**18,3**	3,3	10,1	4,9	18,9	3,6	9	6,3
GS	27,8	6,2	14,5	7,1	25,4	6,2	13,6	5,6
GES	20,2	4,3	8,5	7,3	**17,3**	3,5	8,2	5,6
K2-T	35,4	10,4	15,7	9,3	36,9	12,3	17,4	7,2
K2-R	37,1	11,4	15,1	10,6	40,2	14,6	16,1	9,5
MWST	35,1	4,4	16,3	14,4	34,1	14	16,1	14

Tableau 8.7 – Différences structurelles moyennes entre les réponses des différents algorithmes et la structure du réseau ALARM à partir de 30 bases de cas distinctes. Les valeurs moyennes minimales sont grisées.

mum global – ou l'un deux s'il en existe plusieurs –. Les méthodes évolutionnaires, quant à elles (et en particulier les méthodes telles que AG_{penal} et AG_{dist}), explorent une plus grande partie de l'espace des solutions. Alors que leur avantage premier est de permettre la découverte de plusieurs optima dont, idéalement, l'optimum global, il semble que ce comportement viennent parfois à entraîner la découverte d'optima locaux très bien évalués et pourtant distants, dans l'espace des solutions, de l'optimum recherché.

8.5 Résultats complémentaires

Après avoir observé les résultats des différentes méthodes sur les trois réseaux ASIA, Insurance et ALARM, nous analysons plus en détail le fonctionnement et le comportement des algorithmes génétiques.

8.5.1 Commentaires généraux

Les résultats en terme de scores et de critères graphiques des différentes méthodes nous permettent d'ores et déjà de faire quelques remarques :

Sur les problématiques simples telles que l'apprentissage de la structure *ASIA*, les heuristiques de type glouton comme GS ou GES permettent d'obtenir des résultats de très bonne qualité. Si, d'un point de vue qualitatif, les algorithmes évolutionnaires rivalisent avec, voire dépassent en certaines occasions, les performances de ces méthodes gloutonnes, leurs temps d'exécution sont plus élevés (cf. section 8.6.2).

Avec un nombre de variables restreint, le voisinage de chaque structure dans l'espace des *GOSC* est de petite taille, comparativement à celui d'un réseau tel qu'ALARM. L'emploi de la formule de Robinson (cf. section 4.4) nous permet d'ailleurs de calculer, pour ces deux réseaux, les tailles des espaces de recherche : respectivement près de 8.10^{11} pour ASIA et près de 3.10^{237} pour ALARM.

Par conséquent, l'emploi des algorithmes évolutionnaires devrait se restreindre à l'apprentissage de réseaux complexes et cela quand bien même leur efficacité est vérifiée, qualitativement, sur la recherche de structures simples.

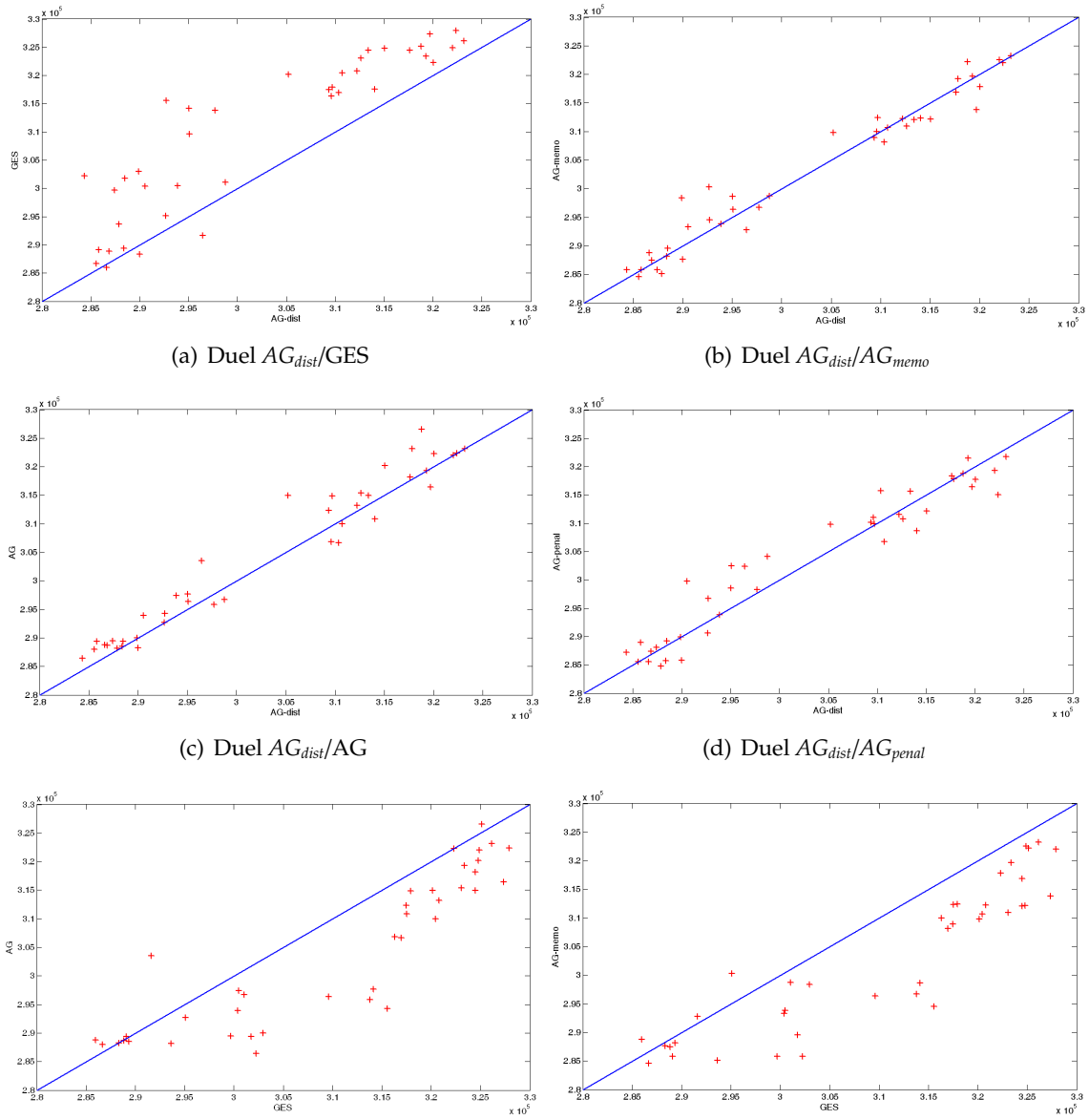
Les méthodes de type K2 et la méthode MWST, malgré plusieurs lancements aléatoires, ne parviennent pas à obtenir de résultats performants sur les réseaux complexes tels qu'Insurance ou ALARM. Si la limitation à l'espace des arbres de l'algorithme MWST explique grandement ses mauvaises performances sur de telles instances, le problème des méthodes de type K2 provient essentiellement de leur dépendance envers l'ordre topologique qui leur est fourni en entrée. Malgré, *a priori*, des initialisations inadéquates et des solutions présentant des distances graphiques à l'original supérieures à celles des solutions renvoyées par les algorithmes évolutionnaires et gloutons, les méthodes K2/MWST obtiennent des scores BIC de bonne qualité bien qu'encore inférieures à celles des heuristiques concurrentes.

8.5.2 Performances

Afin de clarifier les niveaux de performances des algorithmes évolutionnaires non seulement les uns par rapport aux autres mais aussi vis-à-vis de la meilleure heuristique concurrente – GES –, nous avons procédé à l'établissement de confrontations *un contre un* de ces méthodes, sur les deux réseaux complexes Insurance et ALARM.

Ces confrontations ont lieu sur les critères résumés précédemment dans les tables 8.4, 8.5, 8.6 et 8.7 cependant, le détail des performances respectives est ici plus visible.

Les scores BIC mesurés sur les figures 8.4, 8.5, 8.8 et 8.9 correspondent aux scores obtenus sur les bases de vérification des deux réseaux.



(a) Duel AG_{dist}/GES

(b) Duel AG_{dist}/AG_{memo}

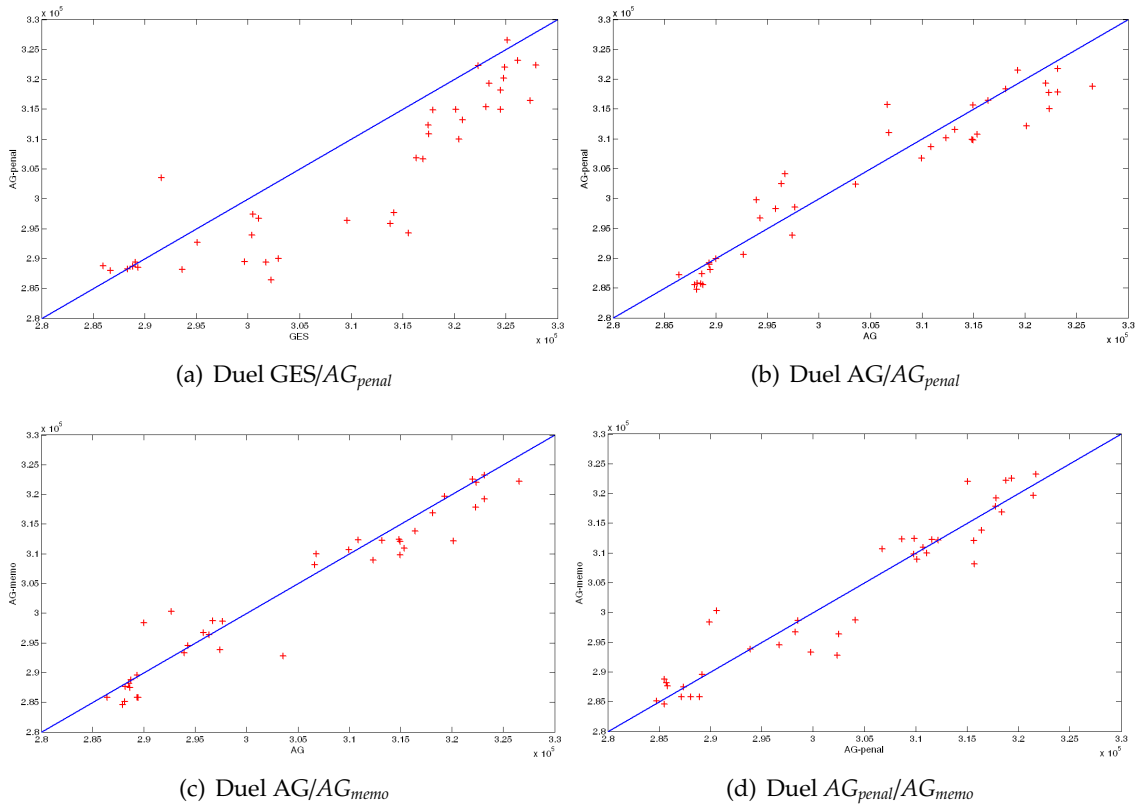
(c) Duel AG_{dist}/AG

(d) Duel AG_{dist}/AG_{penal}

(e) GES/AG

(f) Duel GES/AG_{memo}

Figure 8.4 – Duels inter-méthodes sur l'apprentissage du réseau Insurance. En abscisses et en ordonnées, les opposés des scores BIC des solutions retournées par les méthodes figurant sur les axes correspondants. L'objectif étant de minimiser l'opposé du score, la méthode dont la zone est la moins occupée remporte le duel.



(a) Duel GES/AG_{penal}

(b) Duel AG/AG_{penal}

(c) Duel AG/AG_{memo}

(d) Duel AG_{penal}/AG_{memo}

Figure 8.5 – Duels inter-méthodes sur l'apprentissage du réseau Insurance. En abscisses et en ordonnées, les opposés des scores BIC des solutions retournées par les méthodes figurant sur les axes correspondants. L'objectif étant de minimiser l'opposé du score, la méthode dont la zone est la moins occupée remporte le duel.

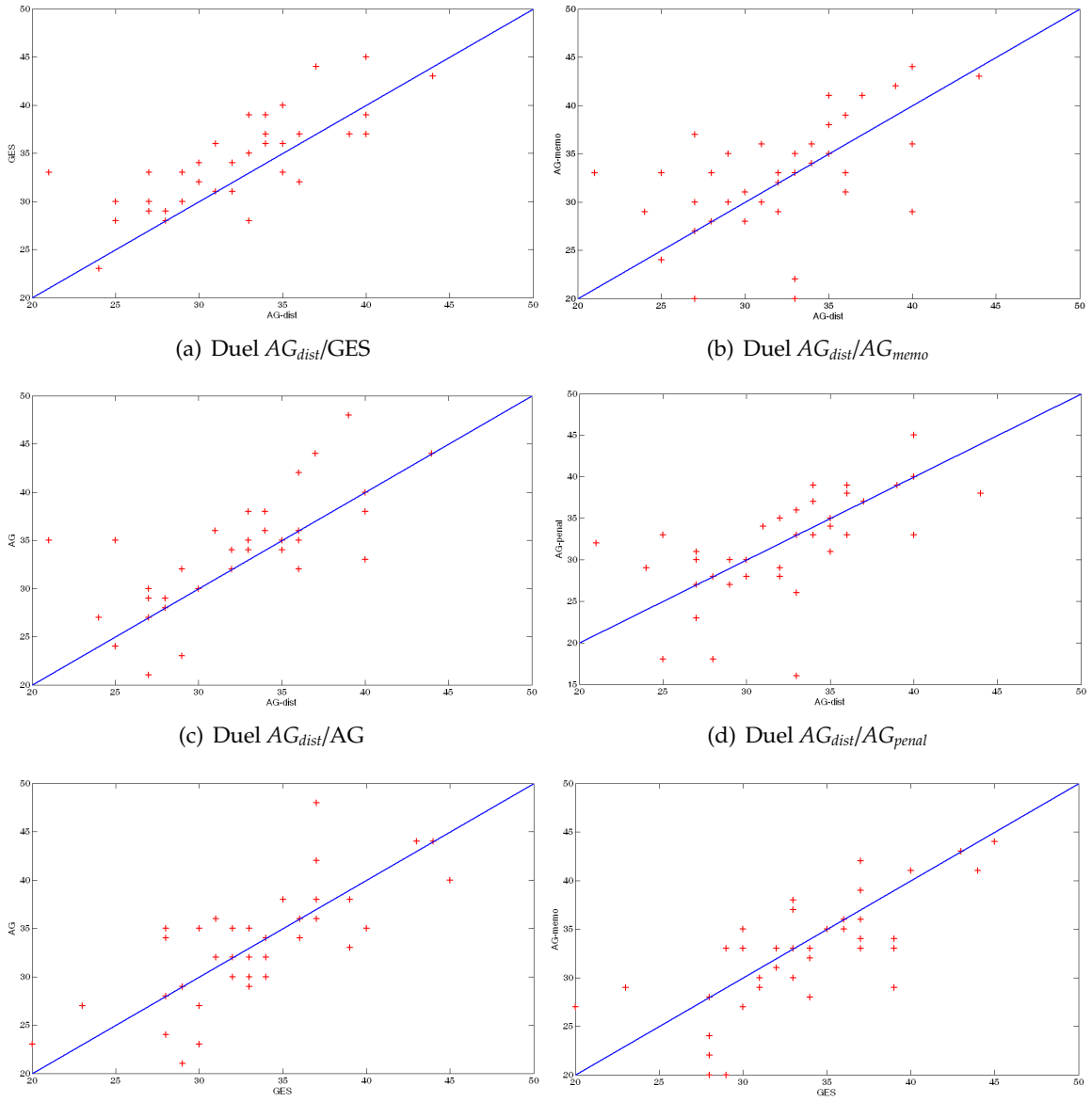
Commentaires

Les résultats des différents duels sur le réseau Insurance permettent de dégager quelques éléments. Il s'avère que l'algorithme distribué AG_{dist} l'emporte clairement au niveau des confrontations sur le score, excepté contre les algorithmes AG_{penal} et AG_{memo} avec lesquels il fait quasiment jeu égal (nombreux points sur ou proches de la bissectrice). L'autre élément remarquable est que l'algorithme GES perd systématiquement ses duels contre les méthodes évolutionnaires, y compris contre l'algorithme simple AG.

Ce phénomène est confirmé par les duels sur la distance d'édition où l'algorithme AG_{penal} trouve certaines structures très proches, étant donné la taille de la base d'apprentissage, du réseau d'origine ; ceci provoquant d'ailleurs sur le graphe un décalage de la bissectrice. Les distances d'édition sont cependant moins lisibles que les performances des scores obtenues et ce en raison d'une plus grande dispersion des nuages de points.

Les méthodes évolutionnaires sont, ici aussi, performantes pour la recherche de cette structure, comparativement aux algorithmes GES et GS. On peut tout de fois remarquer que les confrontations basées sur les différences graphiques ne permettent pas réellement de distinguer les performances entre les algorithme AG et AG_{memo} d'un côté et GES de l'autre. Cependant, ces différences consistent essentiellement en des inversions d'arcs (dont l'impact réel sur la qualité des solutions est difficile à estimer), phénomène généralement évité par GES du fait de sa procédure de construction graduelle de sa solution.

Entre elles, les méthodes évolutionnaires font jeu égal hormis l'algorithme AG dont les performances demeurent inférieures aux autres.



(a) Duel AG_{dist}/GES

(b) Duel AG_{dist}/AG_{memo}

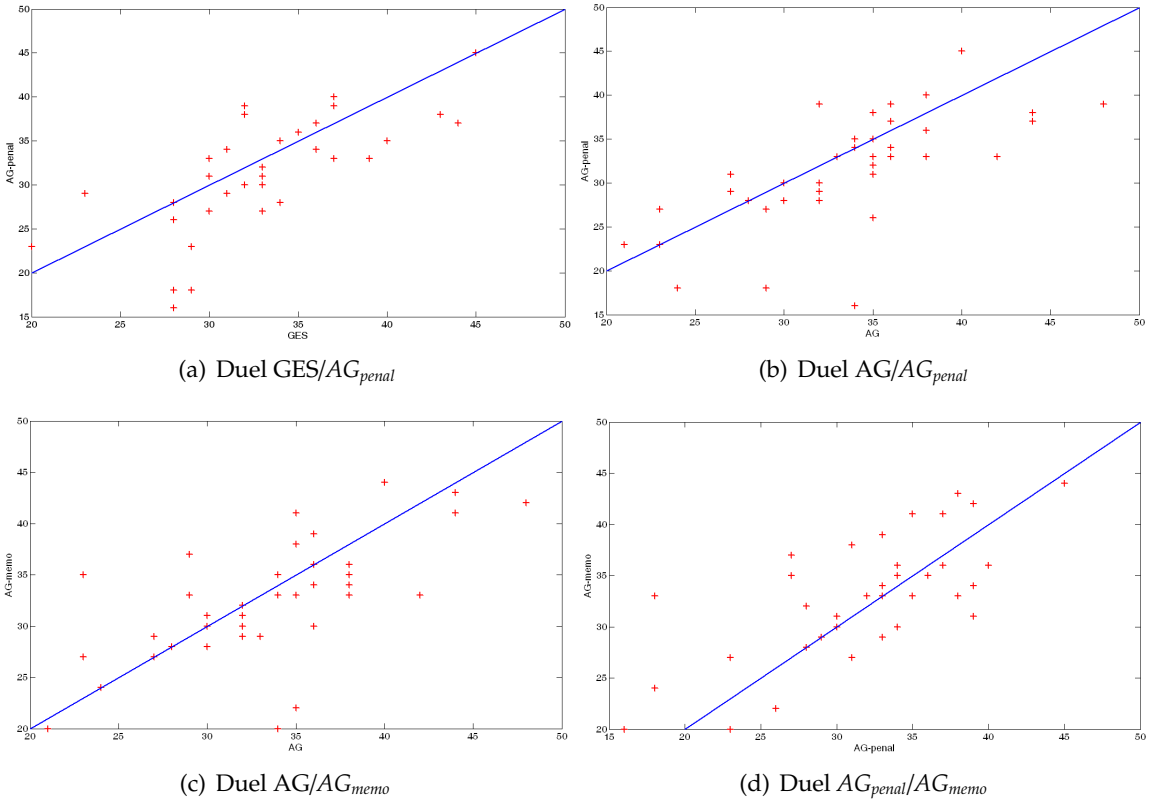
(c) Duel AG_{dist}/AG

(d) Duel AG_{dist}/AG_{penal}

(e) GES/AG

(f) Duel GES/AG_{memo}

Figure 8.6 – Duels inter-méthodes sur l'apprentissage du réseau Insurance. En abscisses et en ordonnées, les mesures des distances graphiques des solutions retournées par les méthodes figurant sur les axes correspondants par rapport au graphe d'origine. L'objectif étant de minimiser la distance d'édition, la méthode dont la zone est la moins occupée remporte le duel.



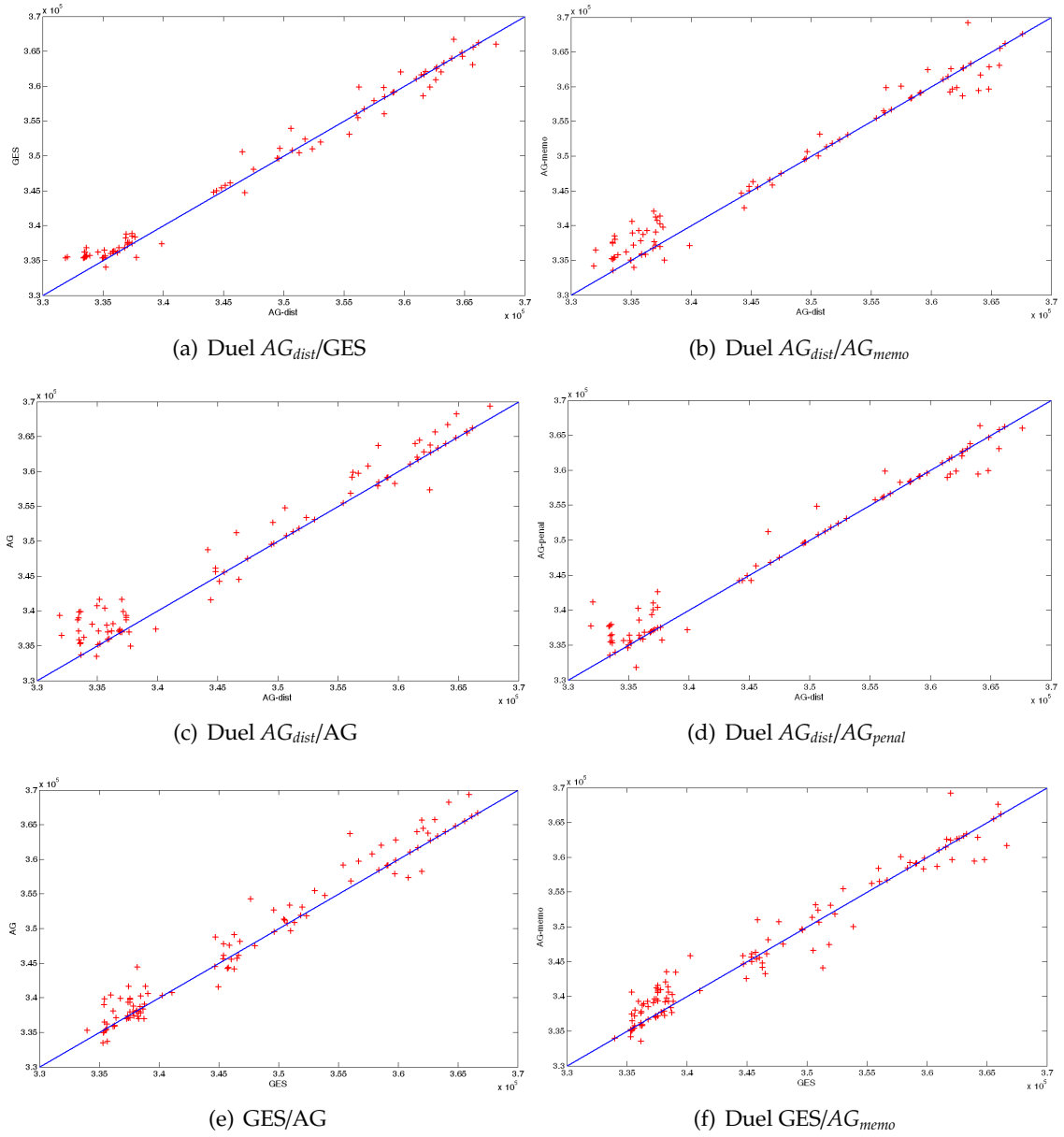
(a) Duel GES/AG_{penal}

(b) Duel AG/AG_{penal}

(c) Duel AG/AG_{memo}

(d) Duel AG_{penal}/AG_{memo}

Figure 8.7 – Duels inter-méthodes sur l'apprentissage du réseau Insurance. En abscisses et en ordonnées, les mesures des distances graphiques des solutions retournées par les méthodes figurant sur les axes correspondants par rapport au graphe d'origine. L'objectif étant de minimiser la distance d'édition, la méthode dont la zone est la moins occupée remporte le duel.



(a) Duel AG_{dist}/GES

(b) Duel AG_{dist}/AG_{memo}

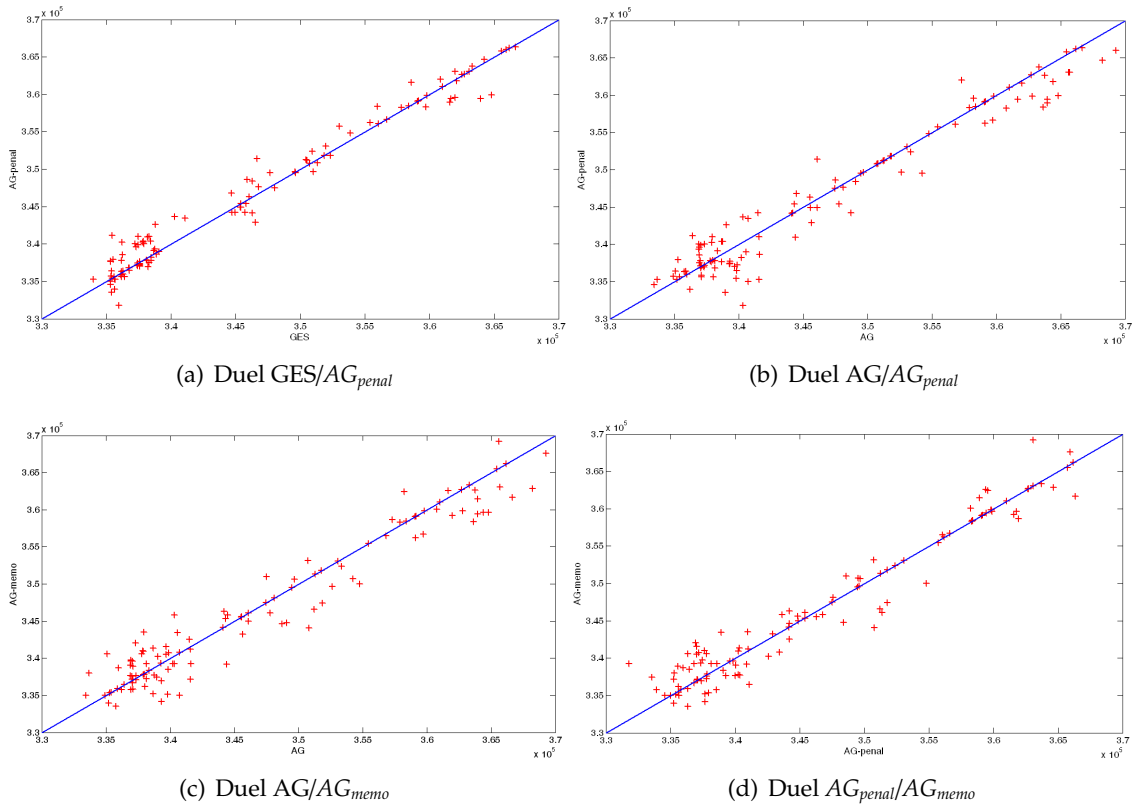
(c) Duel AG_{dist}/AG

(d) Duel AG_{dist}/AG_{penal}

(e) GES/AG

(f) Duel GES/AG_{memo}

Figure 8.8 – Duels inter-méthodes sur l'apprentissage du réseau ALARM. En abscisses et en ordonnées, les opposés des scores BIC des solutions retournées par les méthodes figurant sur les axes correspondants. L'objectif étant de minimiser l'opposé du score, la méthode dont la zone est la moins occupée remporte le duel.



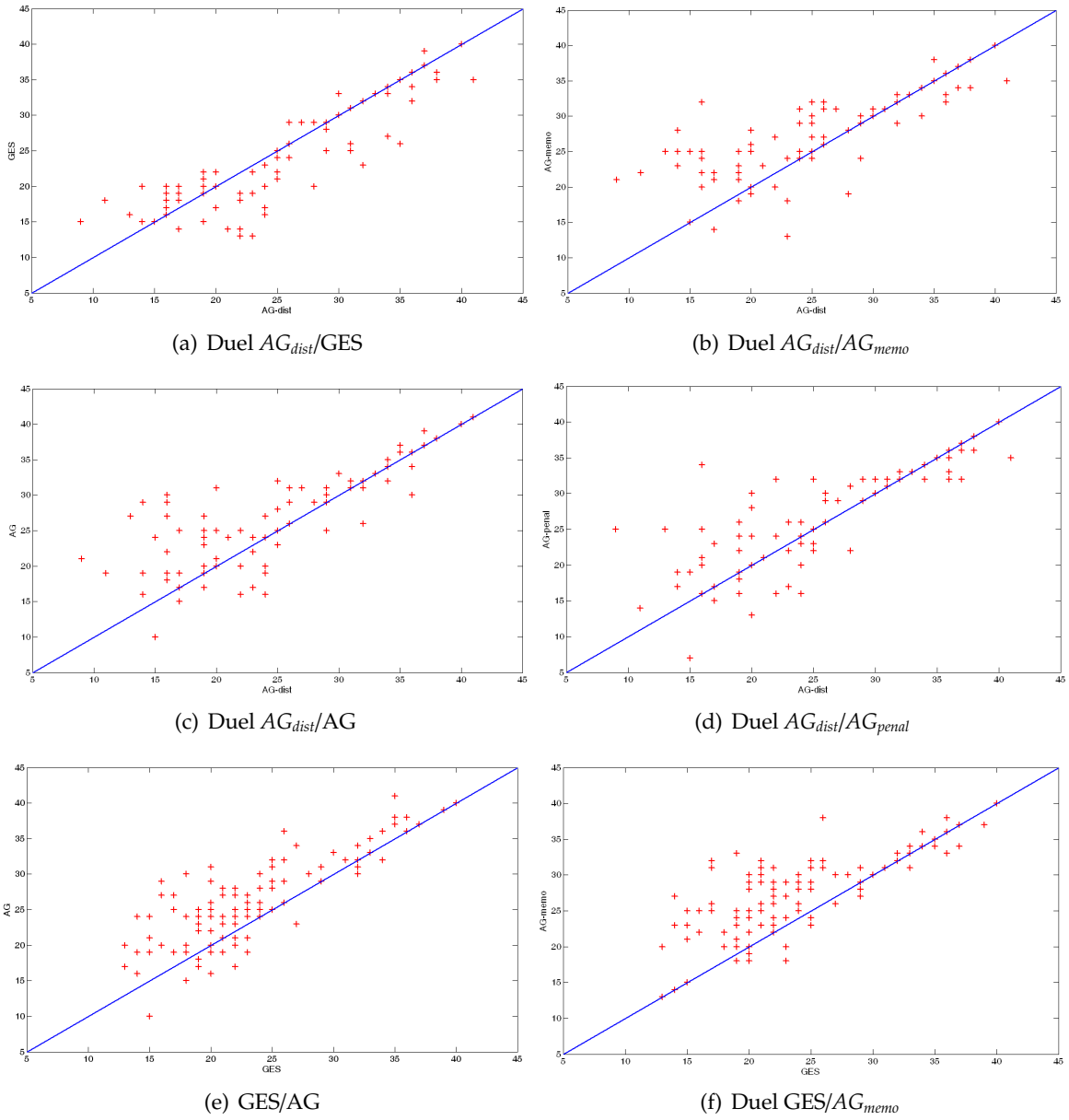
(a) Duel GES/AG_{penal}

(b) Duel AG/AG_{penal}

(c) Duel AG/AG_{memo}

(d) Duel AG_{penal}/AG_{memo}

Figure 8.9 – Duels inter-méthodes sur l'apprentissage du réseau ALARM. En abscisses et en ordonnées, les opposés des scores BIC des solutions retournées par les méthodes figurant sur les axes correspondants. L'objectif étant de minimiser l'opposé du score, la méthode dont la zone est la moins occupée remporte le duel.



(a) Duel AG_{dist}/GES

(b) Duel AG_{dist}/AG_{memo}

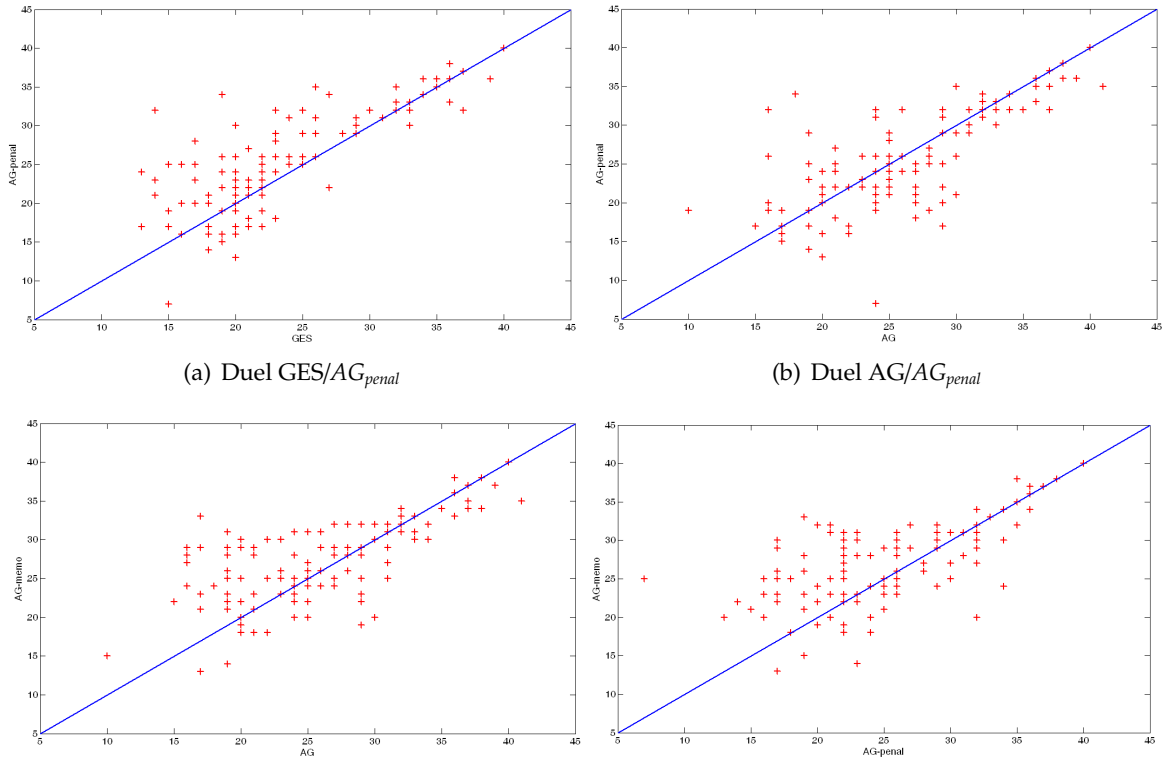
(c) Duel AG_{dist}/AG

(d) Duel AG_{dist}/AG_{penal}

(e) GES/AG

(f) Duel GES/AG_{memo}

Figure 8.10 – Duels inter-méthodes sur l'apprentissage du réseau ALARM. En abscisses et en ordonnées, les mesures des distances graphiques des solutions retournées par les méthodes figurant sur les axes correspondants par rapport au graphe d'origine. L'objectif étant de minimiser la distance d'édition, la méthode dont la zone est la moins occupée remporte le duel.



(a) Duel GES/AG_{penal}

(b) Duel AG/AG_{penal}

(c) Duel AG/AG_{memo}

(d) Duel AG_{penal}/AG_{memo}

Figure 8.11 – Duels inter-méthodes sur l'apprentissage du réseau ALARM. En abscisses et en ordonnées, les mesures des distances graphiques des solutions retournées par les méthodes figurant sur les axes correspondants par rapport au graphe d'origine. L'objectif étant de minimiser la distance d'édition, la méthode dont la zone est la moins occupée remporte le duel.

Commentaires

Les confrontations basées sur le score BIC, avec le réseau ALARM, permettent de dégager deux algorithmes : l'algorithme GES mais aussi l'algorithme distribué AG_{dist}. En concordance avec Les tables 8.7 et 8.6, il s'avère que, pour l'apprentissage de la structure du réseau ALARM, l'algorithme à populations distribuées est le seul pouvant rivaliser, en termes de performances, avec l'algorithme GES. On peut néanmoins remarquer que, ici aussi, bien que l'algorithme AG_{dist} trouve des solutions obtenant un meilleur score, l'algorithme GES trouve en moyenne des solutions graphiquement plus proches du réseau d'origine – d'où un éventuel phénomène de surapprentissage dans le cas de AG_{dist}.

Les autres méthodes évolutionnaires sont difficilement départageables, tant au niveau du score que de la distance d'édition excepté pour l'algorithme adaptatif AG_{memo}. Ce dernier, pour des raisons envisagées dans la section 8.4, renvoie des solutions inférieures à celles des autres méthodes évolutionnaires (ainsi qu'à celles de l'algorithme GES).

8.6 Comportement des algorithmes évolutionnaires

Nous commentons ici diverses mesures propres aux algorithmes évolutionnaires afin, notamment, de pouvoir tirer quelques avis et conclusions quant aux mérites des diverses implémentations.

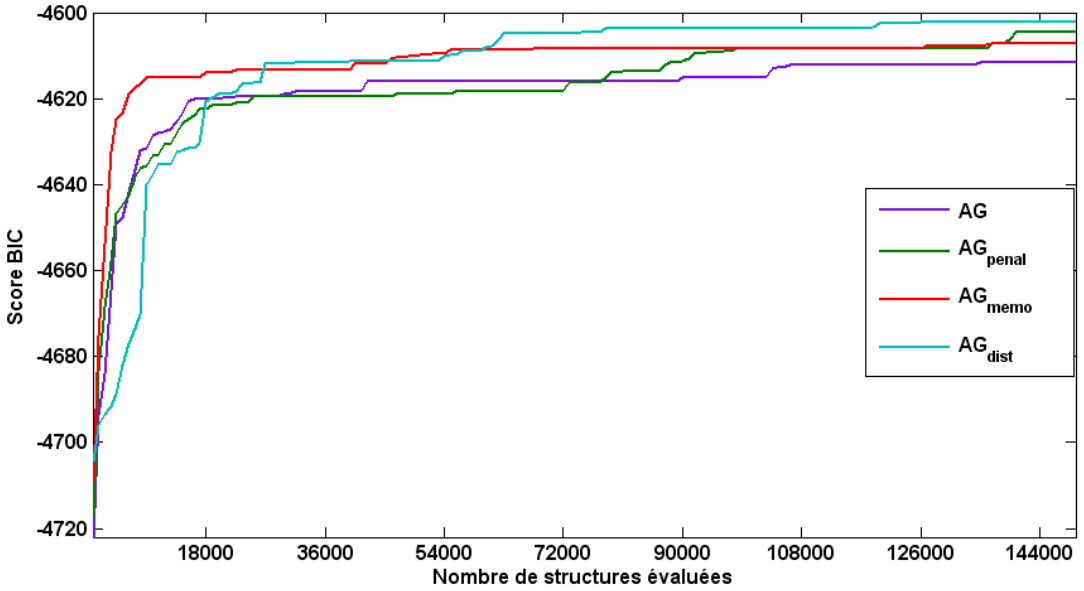
8.6.1 Évolution des individus

Avant de nous intéresser aux figures, il est important de préciser que les courbes des algorithmes AG_{penal} et AG_{dist} ont été "lissées" dans le sens où elles illustrent la valeur du meilleur individu rencontré jusqu'au point considéré. La nature même de la politique de pénalisation de ces méthodes a pour résultat une courbe "en montagnes russes" difficilement lisible.

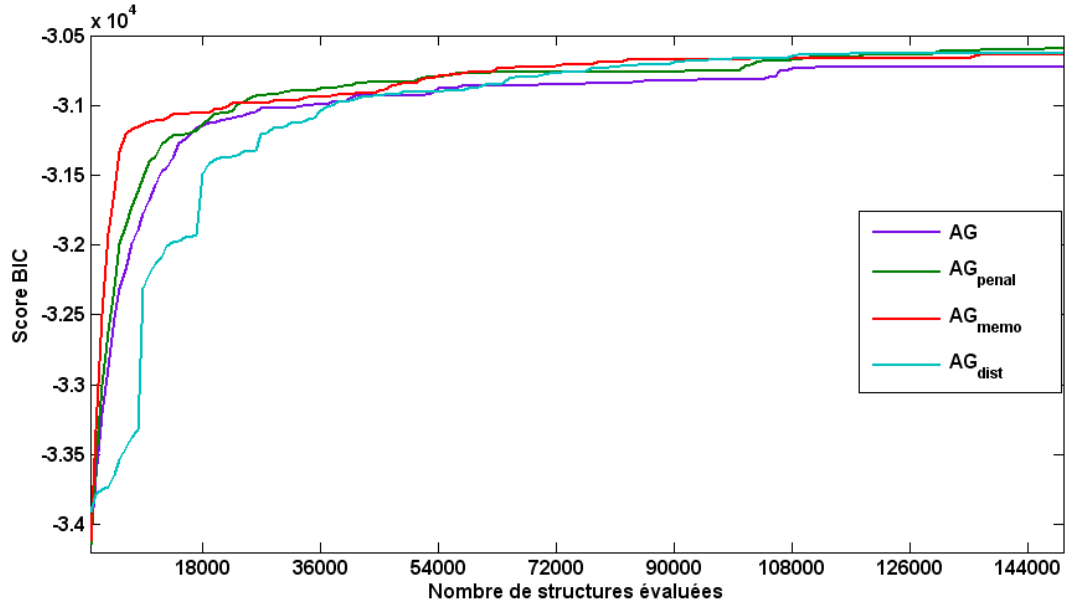
Les courbes correspondent aux performances des différents algorithmes évolutionnaires sur l'apprentissage des structures les plus complexes, celles des réseaux Insurance et ALARM, et ce, pour des bases d'apprentissage de 250 et 2000 cas, respectivement. Nous nous limitons à l'illustration de ces quatre séries d'apprentissage pour les raisons suivantes :

- les apprentissages effectués sur la structure ASIA n'ont pas permis de dégager un comportement caractéristique de la part d'une de nos méthodes par rapport aux autres. Cet exemple est en fait trop limité et ne présente pas de réel intérêt ;
- les séries choisies ici sont duales et permettent de représenter le comportement des méthodes dans le cadre de l'apprentissage de modèles complexes, suivant que la quantité d'information à notre disposition soit très limitée (250 cas par base) ou plus conséquente (2000 cas par base).

Si nous considérons les évolutions sur les bases de tailles les plus restreintes, nous remarquons tout d'abord la performance de l'algorithme distribué AG_{dist}. Celui-ci converge certes plus lentement, lors des premières évaluations, que ses vis-à-vis mais permet d'obtenir, au final et dans la plupart des cas, la meilleure solution au terme d'un même nombre d'évaluations.

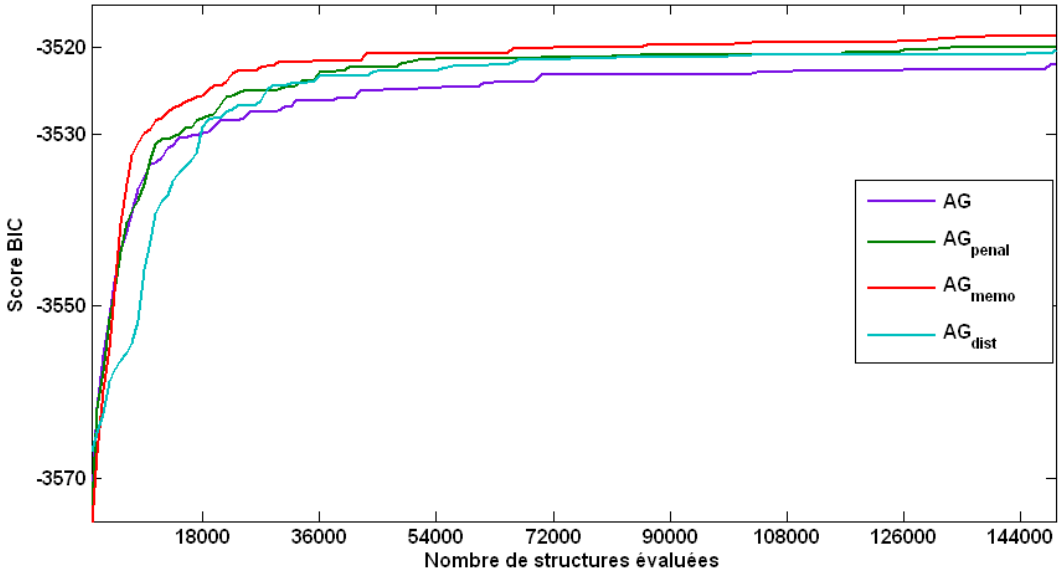


(a) Valeurs des *fitness* pour Insurance, 250 cas.

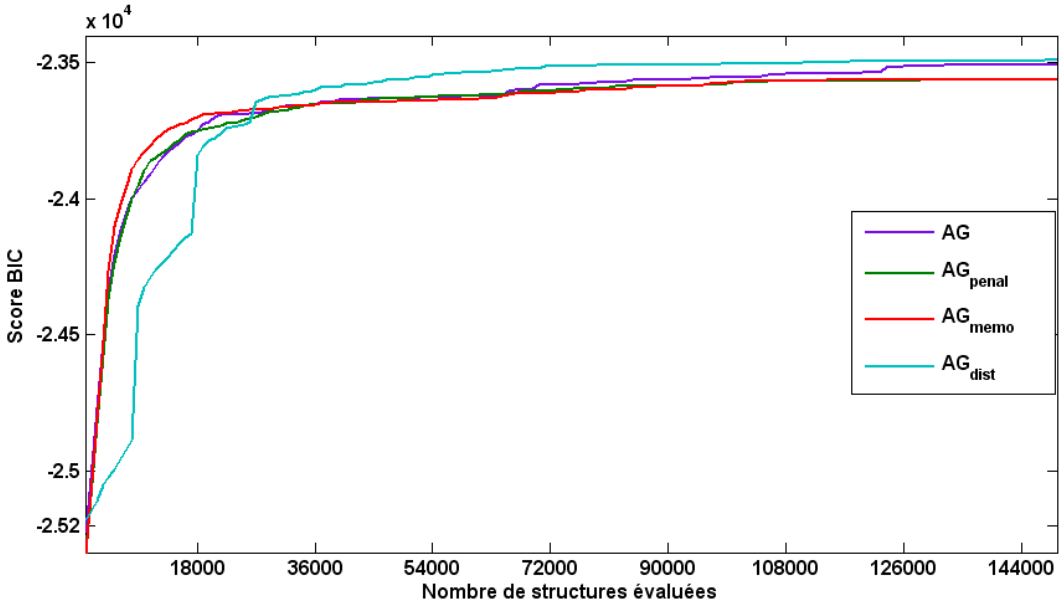


(b) Valeurs des *fitness* pour Insurance, 2000 cas.

Figure 8.12 – Valeurs moyennes des *fitness* en fonction du nombre de structures évaluées pour le réseau Insurance, pour des bases de 250 et 2000 cas, respectivement.



(a) Valeurs des *fitness* pour ALARM, 250 cas.



(b) Valeurs des *fitness* pour ALARM, 2000 cas.

Figure 8.13 – Valeurs moyennes des *fitness* en fonction du nombre de structures évaluées pour le réseau ALARM, pour des bases de 250 et 2000 cas, respectivement.

Les algorithmes disposant d'une stratégie de parcours ou d'adaptation renvoient de meilleures performances que l'algorithme simple *AG* sauf dans le cas de l'apprentissage de la structure du réseau ALARM avec 2000 cas, où cet algorithme, sans toutefois égaler AG_{dist}, évolue progressivement vers une solution plus performante que les autres algorithmes. Ce constat mérite cependant notre attention.

Un retour vers les tables 8.7 et 8.6 montre, par exemple, que les résultats retournés par *AG* ont sensiblement les mêmes performances sur le score BIC que l'algorithme AG_{penal}. Cependant, la distance d'édition des solutions de AG_{penal} demeure inférieure à celle des solutions de *AG*. Qu'en conclure ?

Pour rappel, les figures 8.12 et 8.13 consistent en une moyenne des *fitness* obtenues avec les bases d'apprentissage. Si l'algorithme *AG* est capable d'obtenir des solutions localement très performantes comportant cependant des arcs supplémentaires, on peut sans doute en conclure à un phénomène de surapprentissage débouchant sur un optimum local. Les algorithmes "améliorés", quant à eux, semblent moins sensibles à ce problème en particulier l'algorithme distribué, capable de trouver des solutions performantes à la fois localement, sur les bases d'apprentissage, et globalement, sur la base représentative de 30000 cas.

Une autre observation est que si les algorithmes AG_{penal}, AG_{memo} et AG_{dist} semblent capables, en présence d'un nombre restreint de données, d'effectuer un parcours efficace de l'espace de recherche vers une solution de qualité, les différences entre les diverses variantes ainsi que l'algorithme *AG* semblent s'effacer à mesure que la quantité d'information à disposition augmente.

Lorsque la base de données est limitée, le paysage de la *fitness* contient de nombreux optima locaux, chacun représentant une structure vraisemblable par rapport aux données. Lorsque la taille des bases d'apprentissage augmente, les différentes opportunités, autres que le réseau d'origine, s'effacent. Il paraît donc normal, voire rassurant, de constater qu'en présence d'un choix de plus en plus limité parmi les bonnes solutions présentes dans l'espace de recherche, nos différentes heuristiques sont capables de s'accorder sur la direction à prendre.

Si l'on peut s'accorder sur le fait que les différentes méthodes tendent en général, en fonction de l'information disponible, vers des solutions de qualité, un autre facteur d'intérêt dans notre évaluation est de pouvoir évaluer la vitesse à laquelle cette convergence s'effectue.

8.6.2 Performances temporelles

Les temps indiqués dans cette partie sont donnés à titre indicatif ; sous réserve d'optimisation du code d'une part et/ou de modification de la plate-forme de développement (Matlab).

La vitesse d'exécution est en général reconnue comme étant un des points faibles des méthodes évolutionnaires. Au regard des données des tables 8.8, 8.9 et 8.10, faisant état des temps moyens d'exécution pour l'apprentissage des trois réseaux ASIA, Insurance et ALARM, ceci est vérifié pour nos quatre méthodes avec des temps d'exécution fréquemment supérieurs à ceux, par exemple, des méthodes gloutonnes.

Ces tables recensent les temps d'exécution moyens observés durant nos tests. Rappelons que ces temps correspondent, pour les algorithmes, aux conditions décrites en section 8.1.4 (donc,

par exemple, pour $5 \times n$ instanciations avec ordre topologique aléatoire pour l'algorithme K2-R). Les algorithmes, en particulier pour l'apprentissage de la structure du réseau ASIA, ont été employés dans des conditions différentes ; par conséquent, les temps affichés ne le sont qu'à titre indicatif vis-vis des performances préalablement observées et non comme un réel comparatif de performances entre les méthodes, suivant les cas d'application.

	Réseau ASIA			
	250	**500**	**1000**	**2000**
AG	224 ± 5	236 ± 3	252 ± 2	261 ± 4
AG_{penal}	224 ± 3	239 ± 4	246 ± 2	261 ± 4
AG_{memo}	147 ± 3	152 ± 5	158 ± 2	169 ± 4
AG_{dist}	248 ± 4	256 ± 5	266 ± 2	284 ± 3
GS	29 ± 2	35 ± 3	38 ± 6	45 ± 8
GES	4 ± 0,3	4 ± 0,2	5 ± 0,2	6 ± 0,2
K2-T	4 ± 0,2	4 ± 0,3	6 ± 0,4	6 ± 0,1
K2-R	27 ± 1.2	29 ± 1	30 ± 0,4	35 ± 0,5
MWST	< 1	< 1	< 1	< 1

Tableau 8.8 – Moyennes et écart-types arrondis des temps d'exécution totaux, en secondes, requis pour les différentes méthodes pour l'apprentissage de la structure du réseau ASIA, selon la taille de la base d'apprentissage employée.

	Réseau Insurance			
	250	**500**	**1000**	**2000**
AG	2637 ± 38	2741 ± 46	3010 ± 103	3393 ± 121
AG_{penal}	3366 ± 42	3471 ± 31	3712 ± 93	3950 ± 82
AG_{memo}	2842 ± 51	3000 ± 43	3484 ± 32	4012 ± 126
AG_{dist}	7148 ± 122	7345 ± 92	7553 ± 205	7977 ± 220
GS	1281 ± 152	1395 ± 172	1809 ± 230	3327 ± 530
GES	642 ± 69	757 ± 74	1203 ± 51	2007 ± 228
K2-T	252 ± 4	299 ± 7	324 ± 2	411 ± 8
K2-R	1325 ± 23	1496 ± 16	1789 ± 23	2188 ± 20
MWST	< 5	< 5	< 5	< 5

Tableau 8.9 – Moyennes et écart-types arrondis des temps d'exécution, en secondes, requis pour les différentes méthodes pour l'apprentissage de la structure du réseau Insurance, selon la taille de la base d'apprentissage employée.

Les temps de calcul des algorithmes MWST et K2-T sont très courts, sur les réseaux Insurance et ALARM, comparativement aux méthodes de type *AG* ou gloutonnes. MWST limite cependant sa recherche que sur l'espace des arbres tandis que la méthode K2-T accepte en entrée un ordre topologique (fourni par MWST) au regard duquel la qualité des résultats fournis est très sensible.

La variante K2-R prend, quant à elle, un temps d'autant plus grand que la taille du problème implique un grand nombre d'instances de l'algorithme K2, chacune d'entre elles aléatoirement initialisée. Cette approche semble d'autant plus vaine que les performances de l'algorithme K2-T sont tout à fait comparables pour un temps d'exécution dérisoire.

On peut cependant remarquer que les temps d'exécution des méthodes évolutionnaires,

	Réseau ALARM			
	250	**500**	**1000**	**2000**
AG	3593 ± 47	3659 ± 41	3871 ± 53	4088 ± 180
AG_{penal}	3843 ± 58	3877 ± 44	4051 ± 59	4332 ± 78
AG_{memo}	3875 ± 32	4005 ± 43	4481 ± 46	4834 ± 52
AG_{dist}	9118 ± 269	9179 ± 285	9026 ± 236	9214 ± 244
GS	9040 ± 1866	9503 ± 1555	12283 ± 1403	16216 ± 2192
GES	3112 ± 321	2762 ± 166	4055 ± 3, 4	5759 ± 420
K2-T	733 ± 9	855 ± 25	1011 ± 14	1184 ± 8
K2-R	3734 ± 61	4368 ± 152	5019 ± 67	5982 ± 43
MWST	10 ± 1	10 ± 2	11 ± 1	12 ± 1

Tableau 8.10 – Moyennes et écart-types arrondis des temps d'exécution totaux, en secondes, requis pour les différentes méthodes pour l'apprentissage de la structure du réseau ALARM, selon la taille de la base d'apprentissage employée.

bien qu'importants, demeurent relativement stables avec l'augmentation de la complexité du problème à traiter. Ceci est particulièrement remarquable avec les apprentissages effectués sur le réseau ALARM. Dans ce dernier cas, le nombre de variables pour chaque graphe exploité définit un voisinage particulièrement large. Alors que l'algorithme génétique procède par une approche *generate and test* guidée par la performance des éléments de la population, les algorithmes gloutons effectuent un parcours exhaustif de ce voisinage et voient donc leurs temps d'exécution augmenter radicalement.

À noter que la version de l'algorithme GS qui a été testée emploie un cache afin d'éviter le recalcul de certains scores. Mais en présence d'un voisinage important, le parcours même d'un cache de taille (trop) importante prend lui aussi, au final, un temps conséquent.

Dès lors que la recherche de la structure atteint une certaine complexité (ALARM avec une base d'apprentissage de 2000 cas), les algorithmes évolutionnaires deviennent plus rapides que les méthodes gloutonnes.

8.6.3 Nombre d'itérations avant la solution

Un facteur important est le nombre d'itérations requises par les différents algorithmes génétiques pour trouver leur solution. Ces résultats figurent dans la table 8.11.

Il est important de parler ici de nombre d'itérations avant de trouver le meilleur individu et non pas de nombre d'itérations avant convergence, comme cela est le cas dans la littérature. En effet, la stratégie de *niching* séquentiel mise en place au sein des algorithmes AG_{penal} et AG_{dist} introduit, ponctuellement et régulièrement, des perturbations consécutives à la modification de la *fitness*, interdisant à la population de converger.

La lecture du nombre moyen d'itérations avant obtention du meilleur individu – meilleur individu sur le nombre total d'itérations prédéfini par l'utilisateur – est une valeur dont il est malheureusement assez difficile de retranscrire la signification. Bien qu'il soit naturel de penser qu'un algorithme retournant sa meilleure solution en un temps très bref soit préférable, nous

ne devons pas négliger le fait que, pour un algorithme évolutionnaire, une telle rapidité peut aussi être synonyme d'une convergence prématurée vers un optimum local. L'inverse est aussi vrai si l'algorithme prend trop de temps pour trouver sa meilleure solution.

	Réseau ASIA			
	250	**500**	**1000**	**2000**
AG	14 ± 14	31 ± 23	23 ± 15	30 ± 23
AG_{penal}	16 ± 16	33 ± 29	26 ± 24	38 ± 29
AG_{memo}	13 ± 12	15 ± 8	17 ± 11	21 ± 4
AG_{dist}	15 ± 12	15 ± 8	17 ± 11	22 ± 19
	Réseau Insurance			
	250	**500**	**1000**	**2000**
AG	364 ± 319	454 ± 295	425 ± 249	555 ± 278
AG_{penal}	704 ± 295	605 ± 321	694 ± 258	723 ± 234
AG_{memo}	398 ± 326	414 ± 277	526 ± 320	501 ± 281
AG_{dist}	82 ± 59	106 ± 77	166 ± 84	116 ± 27
	Réseau ALARM			
	250	**500**	**1000**	**2000**
AG	265 ± 257	417 ± 271	552 ± 244	529 ± 245
AG_{penal}	380 ± 291	535 ± 225	640 ± 262	624 ± 253
AG_{memo}	341 ± 269	474 ± 332	592 ± 291	592 ± 249
AG_{dist}	87 ± 76	141 ± 90	212 ± 66	186 ± 72

Tableau 8.11 – Moyennes et écart-types arrondis des nombres d'itérations nécessaires à chaque méthode pour obtenir sa meilleure proposition de solution.

Si l'on croise les données de la table 8.11 avec celles de la table 8.10, on peut se rendre compte que le ratio des temps d'exécution des algorithmes et en particulier ceux d'AG_{dist} et du nombre d'itérations moyen mis par ces algorithmes pour trouver leurs meilleures solutions, la vitesse de nos différentes méthodes devient compétitive avec celle d'algorithmes tels que GS et GES.

8.6.4 Taux d'individus réparés

Comme nous l'avons précisé dans la section 6.1.5, l'ensemble de nos méthodes évolutionnaires emploient un opérateur de réparations, basé sur l'emploi de l'information mutuelle entre chaque paire de variables, afin de supprimer les circuits pouvant éventuellement apparaître dans un individu au cours de l'évolution.

Ces circuits ne peuvent en réalité apparaître qu'à l'issue de la phase de mutation, soit par ajout soit par inversion d'un arc. En effet, l'opérateur de croisement, défini dans la section 6.1.5, présente la particularité d'être fermé par rapport à l'espace des *GOSC* (i.e. tout transfert d'un ensemble de parents créant un circuit chez le descendant sera ignoré).

Nous avons mesuré le taux moyen d'individus réparés (par rapport au nombre d'individus générés à chaque itération de l'algorithme) au cours d'instances de nos différentes méthodes. Les paramètres des algorithmes sont les mêmes que ceux employés au cours de nos tests et les résultats présentés dans la table 8.6.4 correspondent à une moyenne sur dix instances de chaque algorithme, pour un réseau et une taille donnée de base d'apprentissage.

	Taux de réparations - ASIA			
	250	**500**	**1000**	**2000**
AG	13%	15,2%	15,3%	16,5%
AG_{penal}	13,6%	15,3%	15,9%	16,5%
AG_{memo}	12,4%	15,7%	15,8%	16,1%
AG_{dist}	13,7%	15,3%	14,5%	15,6%

	Taux de réparations - Insurance			
	250	**500**	**1000**	**2000**
AG	11%	13,2%	13,9%	15,1%
AG_{penal}	13,4%	14,7%	15,9%	15,3%
AG_{memo}	12,3%	13,9%	13,6%	14,9%
AG_{dist}	9,4%	10,3%	12,6%	14,4%

	Taux de réparations - ALARM			
	250	**500**	**1000**	**2000**
AG	6,1%	5,2%	6,4%	7%
AG_{penal}	4,6%	5,3%	5,6%	6,5%
AG_{memo}	5,7%	6,2%	6,3%	8,4%
AG_{dist}	5,5%	5,1%	6,8%	7,1%

Tableau 8.12 – Taux d'individus ayant recouru à l'opérateur de réparation, pour les différentes méthodes évolutionnaires selon le réseau appris et la taille de la base d'apprentissage.

On peut remarquer, au vu de la table 8.6.4, que le taux moyen d'individus réparés ne semble pas dépendre de la méthode adoptée mais bien :

1. du réseau appris et de sa complexité ;
2. de la quantité d'information disponible à l'apprentissage.

Ainsi, le réseau ASIA, présentant pourtant la structure la plus simple parmi celles sur lesquelles nous avons effectué nos tests, présente le taux le plus important d'individus réparés (à peu près 15%). Ces résultats ne dépendent pas seulement d'une convergence rapide de la population au voisinage d'un point, puisque le réseau Insurance, plus complexe et nécessitant plus de recherche, implique un taux de réparations moyen à peine inférieur à celui constaté avec le réseau ASIA. En revanche, les apprentissages effectués sur le réseau ALARM n'impliquent que des taux avoisinants les 5%. Les caractéristiques des réseaux sont directement en cause.

En ce qui concerne l'impact de ces réparations sur les calculs effectués : une réparation n'implique de calculs de score, localement en une variable, que dans la mesure où la suppression d'arc visant à éliminer le circuit créé efface un arc différent de celui venant d'être créé, par ajout ou par inversion. Dans le cas contraire, l'individu et sa *fitness* (y compris les scores locaux) sont remis à leur état d'origine.

Considérant ceci, un taux moyen de réparations de l'ordre de 5 à 10% (chiffres figurant une borne supérieure pour les calculs liés à la *fitness*, toute réparation n'impliquant pas forcément de ré-évaluation) paraît raisonnable.

8.7 Conclusion

Les différents résultats et analyses effectués au cours de ce chapitre nous ont permis de dégager plusieurs points concernant non seulement les performances de nos méthodes mais

aussi leur utilité.

Les différentes méthodes évolutionnaires améliorées (AG_{penal}, AG_{memo} et AG_{dist}) permettent d'obtenir de très bons résultats, fréquemment meilleurs que ceux renvoyés par l'algorithme GES, en particulier en présence de structures présentant de nombreuses dépendances difficilement détectables (réseau Insurance). La représentativité des bases d'apprentissage employées joue, dans le cadre de cette compétition entre les méthodes évolutionnaires et l'algorithme GES, dans le cas de l'apprentissage de la structure du réseau ALARM où les algorithmes génétiques sont aptes à profiter d'une quantité d'information plus conséquente (bases de 1000 et 2000 cas) tout en proposant des résultats équivalents à ceux de GES pour des bases de taille plus limitée.

Les différentes stratégies développées ont permis d'améliorer le parcours de l'espace de recherche, ces algorithmes dépassant fréquemment en performances les méthodes couramment employées pour l'apprentissage de structures.

Les méthodes évolutionnaires sont-elles préférables aux méthodes usuelles ? Les algorithmes évolutionnaires, dans le cas général, sont relativement "lourds" à mettre en place. Les temps de calcul sont, par exemple, important pour les méthodes évolutionnaires et ce, dès l'apprentissage de structures simples telles qu'ASIA. S'il est vrai que l'on peut espérer des temps de convergence somme toute raisonnables vers la meilleure solution, la nature stochastique des algorithmes génétiques ne peut fournir qu'une estimation du nombre d'itérations à effectuer.

En ce qui concerne l'apport des différentes méthodes que nous avons proposées, seules les méthodes exploitant le concept de *niching* séquentiel permettent une réelle amélioration des performances de l'algorithme génétique. En particulier, la stratégie revenant à combiner répartition spatiale et *niching* séquentiel a permis d'obtenir des résultats équivalents, sinon meilleurs que ceux de l'algorithme GES, dans le cadre de nos tests.

S'il est vrai que notre stratégie d'adaptativité de l'opérateur de mutation a permis d'obtenir, dans certains cas, de très bons résultats, cette méthode n'a pas présenté des résultats de qualité constante à travers les différents apprentissages auxquels nous avons procédé.

Si le principe d'adaptativité que nous avons présenté nous semble être une bonne idée, son application nécessiterait de modifier notre manière d'aborder le problème. De prime abord, un apport essentiel à l'approche par un algorithme génétique de l'apprentissage de structures serait d'opter pour une représentation adaptée. Notre représentation, semblable à celle utilisée par [Larranaga et al., 1996] a pour défaut de conférer une représentation mal adaptée, présentant un trop grand nombre de degrés de liberté pour le processus évolutionnaire. Une possibilité pourrait être la représentation proposée par [van Dijk et Thierens, 2004] où l'algorithme présenté procède à une réduction de l'espace de recherche en procédant à une série de tests statistiques d'ordre faible avant de faire évoluer une population sur la partie de l'espace des structures définie par le squelette restant.

Troisième partie

Réseaux bayésiens : une application à la reconnaissance de formes

Chapitre 9

La segmentation de l'iris dans une image

9.1 Introduction

Il est un domaine où l'emploi des réseaux bayésiens est reconnu : la classification. Si l'on considère une classe comme étant la cause (ou la conséquence) de ses caractéristiques, un modèle prenant en compte ces interactions paraît alors parfaitement à même de permettre la détermination d'une classe à partir d'une base de caractéristiques.

Nous introduisons dans ce chapitre une application des modèles bayésiens en tant que classifieurs à la segmentation de pixels dans un ensemble d'images.

9.2 Réseaux bayésiens pour la classification

La classification est une problématique fréquente dans des domaines tels que le traitement de données ou la reconnaissance de formes. Elle conduit à la construction d'un modèle ou d'un algorithme officiant en tant que classificateur et devant indiquer la classe d'appartenance d'un objet à partir des valeurs prises par un ensemble de ses caractéristiques.

Les réseaux bayésiens, par le biais de l'inférence, notamment, se prêtent particulièrement à cet exercice [Cheeseman et al., 1988]. De nombreux chercheurs ont développé des modèles particuliers de réseaux bayésiens dédiés à la classification. Cette section va présenter trois des modèles les plus répandus : le réseau bayésien naïf, les réseaux augmentés et l'approche par multi-nets. Nous cherchons ici à déterminer la classe X_c d'un objet à partir des valeurs de ses n caractéristiques $\{X_1, X_2, \ldots, X_n\}$.

9.2.1 Réseaux bayésiens naïfs

Les réseaux bayésiens naïfs, décrits dans [Langley et al., 1992], font l'hypothèse de l'indépendance des caractéristiques entre elles et se présentent sous la forme de l'exemple de la figure

9.1.

Il a été fréquemment observé que, même dans les cas où l'indépendance entre les caractéristiques n'est pas vérifiée, le classificateur naïf conserve des performances proches de l'optimal [Domingos et Pazzani, 1996]. Ce modèle, à la fois un des premiers et le plus simple [Ling et Zhang, 2002], renvoie des résultats de bonne qualité et s'avère très robuste. Il a été ainsi employé dans de nombreuses applications pratiques telles que le tri de pourriel [Sahami et al., 1998] ou la classification de texte de manière plus générale [McCallum et Nigam, 1998].

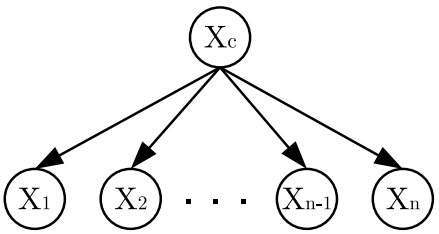


Figure 9.1 – Exemple de réseau bayésien naïf.

9.2.2 Structures arborescentes augmentées

La modélisation naïve présentée précédemment pose l'inconvénient majeur de faire l'hypothèse, rarement vérifiée, d'indépendance entre les différentes caractéristiques. Malgré les performances démontrées dans des cas où cette hypothèse est invalidée [Domingos et Pazzani, 1996], il est certains cas où il peut être préférable de représenter les dépendances régnant entre les caractéristiques [Rish, 2001].

Il est alors possible de conserver une partie de la structure naïve en reliant la variable classe à chacune des caractéristiques tout en permettant l'ajout de liens entre celles-ci.

La structure reliant les caractéristiques peut être quelconque (obtenue par un algorithme d'apprentissage classique) – dans ce cas on parle de BAN (*Bayesian Network Augmented Naive Bayes* – ou bien sous forme d'arborescence auquel cas on obtient un modèle de type TANB (*Tree Augmented Naive Bayes*) [Friedman et al., 1997].

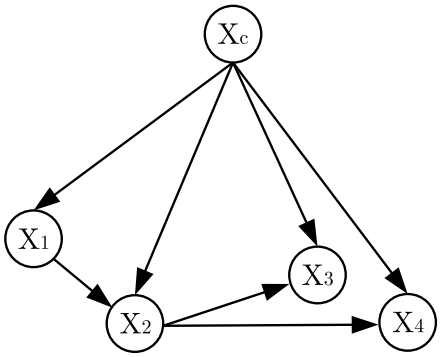


Figure 9.2 – Exemple de réseau bayésien naïf augmenté par un arbre.

9.2.3 Multi-nets

L'approche par multi-nets est un héritage direct de l'approche par réseaux augmentés suivant le principe que les liens existants entre les caractéristiques peuvent varier d'une instance de classe à une autre.

Ici, les différentes classes sont chacune représentées par un réseau dédié constitué des différentes caractéristiques du problème mais dénué de la variable représentant la classe d'appartenance. Pour un problème de classification à n_c classes, nous construisons donc n_c réseaux, chacun comportant uniquement les n variables représentant les caractéristiques. La classification est opérée en élicitant, pour une instance à classifier, la classe liée au réseau ayant la plus grande probabilité jointe $P(X_1, X_2, \ldots, X_n)$.

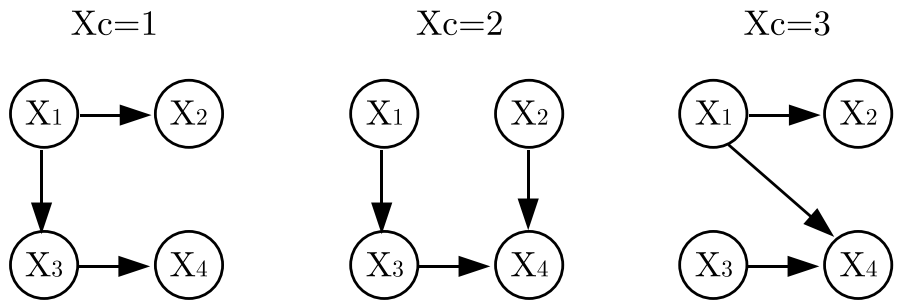


Figure 9.3 – Exemple d'approche par multi-nets pour un problème à trois classes.

9.3 Problématique abordée

Hors de tout propos visant à l'identification biométrique, nous nous intéressons dans ce chapitre aux performances des réseaux bayésiens appliqués au problème de la segmentation de l'iris. Ce dernier problème vise, à partir d'une image de l'œil d'un individu, à distinguer la partie de cette image correspondant à l'iris.

La partie visible de l'œil est composée de plusieurs éléments :

Sclère : membrane blanche et opaque formant le blanc de l'œil ;

Iris : membrane circulaire bordant la pupille. L'iris ne participe à la vision qu'en tant que régulateur de l'illumination de la rétine : ses contractions permettent en effet de contrôler la quantité de lumière pénétrant la pupille ;

Pupille : orifice situé au milieu de l'iris. Noire d'apparence car la lumière y pénètre. Sa taille peut varier en fonction des contractions de l'iris.

Généralement, la pupille se distingue donc comme étant un disque noir central, entouré par l'iris, lui même entouré par la sclère, plus claire. La problème est cependant plus complexe dans le cadre du traitement de l'image car il est nécessaire de prendre en compte plusieurs facteurs générateurs de bruit :

Focus : perte du focus, l'image apparaît floue ;

Réflection : réflection de l'éclairage ambiant, visible en particulier à la surface de la pupille ;

Obstruction : il arrive fréquemment que l'iris soit partiellement masqué par des artefacts tels que les cils ou les paupières.

Un autre problème est la difficulté à distinguer, dans le cas de photographies prises en conditions naturelles, la pupille, noire, d'un iris foncé.

Le problème de la segmentation de l'iris et de l'élimination du bruit a été traité dans de nombreux ouvrages et publications [Daugman, 2007, Monro et al., 2007, Proenca et Alexandre, 2007, He et Shi, 2007] néanmoins nous ne présenterons ici que deux des méthodes les plus répandues.

9.4 Travaux antérieurs

9.4.1 Méthode de J. Daugman

Cette méthode est la plus connue pour plusieurs raisons. Tout d'abord, il s'agit de la méthode pionnière dans le domaine. De plus, cette méthode est la plus robuste à ce jour, ce qui lui a valu d'être la méthode principalement implémentée de par le monde dans les dispositifs commerciaux servant à l'identification par l'iris. John Daugman a de plus développé conjointement une méthode d'identification par détection des indépendances conditionnelles entre les différentes traductions de l'iris en messages binaires.

La méthode de segmentation de l'iris, telle que présentée dans [Daugman, 1993], repose sur l'emploi de l'opérateur intégro-différentiel suivant :

$$max_{r,x_0,y_0} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} ds \right| \tag{9.1}$$

Cet opérateur recherche, sur une image préalablement traitée par un flou gaussien de paramètre σ, les paramètres (r, x_0, y_0) maximisant l'intégrale de l'intensité $I(x, y)$ sur un contour circulaire de centre de coordonnées (x_0, y_0) et de rayon r. Plus simplement, nous recherchons le cercle de centre de coordonnées (x_0, y_0) et de rayon r ayant la dérivée partielle la plus élevée par rapport au rayon r' voisin.

L'efficacité de cette méthode repose intrinsèquement sur la distinction entre l'iris et la pupille d'une part et l'iris et la sclère d'autre part.

9.4.2 Méthode de Wildes

L'auteur de cette méthode, décrite dans [Wildes, 1997], propose de détecter l'iris en utilisant conjointement l'algorithme *Canny Edge* [Canny, 1986], et les transformées de Hough circulaires.

Le détecteur de contours est, dans un premier temps, appliqué à l'image puis, la transformée de Hough circulaire permet de déterminer tout d'abord le cercle correspondant à la frontière entre la sclère et l'iris puis, dans les limites de ce cercle, le cercle formé entre la pupille et l'iris.

Une méthode similaire est proposée par Libor Masek [Masek, 2003] ; celle-ci présente l'avantage de disposer d'une implémentation librement distribuée [Masek et Kovesi., 2003].

9.5 Notre méthode

Nous proposons une méthode consistant à employer un réseau bayésien afin de pouvoir définir si oui ou non un pixel donné de l'image appartient à un iris. Ce travail se place dans le cadre de l'évaluation des réseaux bayésiens en tant que classificateurs et nous ne visons donc pas ici à surpasser les méthodes précitées, en particulier sur le point des temps de calcul. Les méthodes de segmentation de l'iris existantes cherchent en effet toutes à atteindre des temps d'exécution très brefs. Or, nous verrons que les procédés que nous employons sont coûteux en temps de calcul et donc non compétitifs en la matière.

Notre centre d'intérêt est de pouvoir évaluer qualitativement les performances des modèles bayésiens appliqués à la segmentation dans une image.

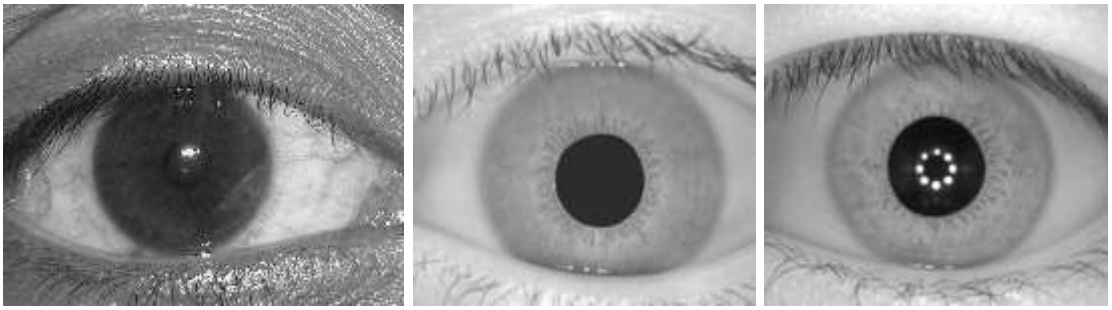
9.5.1 Caractéristiques employées

Nous avons vu que les modèles bayésiens employés dans la classification reposent sur l'inférence de la variable classe à partir des valeurs prises par les caractéristiques. Pour ces dernières, nous avons choisi d'employer des caractéristiques de la texture dans l'image : les caractéristiques d'Haralick. Celles-ci sont présentées dans l'annexe B de même que la notion de matrice de cooccurrence.

9.5.2 La base Ubiris

Cette base d'images [Proença et Alexandre, 2005] consiste en des images prises en conditions réelles. Contrairement aux images couramment employées de bases telles que la base CASIA [Sun, 2006], où les prises de vues sont effectuées par l'intermédiaire d'une caméra infrarouge, les images issues d'Ubiris présentent la plupart des bruits que nous avons cités en section 9.3. Notons cependant qu'une nouvelle base CASIA (nommée CASIA V.3) a vu récemment le jour et comporte, toujours en prises de vues infrarouges, divers bruits parmi lesquels des reflets ou encore des problèmes de positionnement.

Les figures 9.4(a),(b) et (c) présentent des exemples d'images issues de ces bases.



(a) Image de la base Ubiris. (b) Image de la base CASIA V.1. (c) Image de la base CASIA V.3.

Figure 9.4 – Images d'iris issues de bases différentes.

9.6 Les modèles employés

Le modèle bayésien naïf (cf section 9.2.1), reconnu pour sa robustesse et ses performances générales, nous a paru être un modèle de choix pour cette application.

Nous avons cependant employé deux modèles naïfs distincts. Le premier modèle, dénommé simplement *NB*, est un modèle naïf constitué de la variable classe pouvant prendre les valeurs *Iris* ou $\overline{Iris}$ et des caractéristiques de texture d'Haralick.

Un deuxième modèle, dénoté B_s, revient à une architecture similaire à celle de *NB* mais avec un nombre réduit de caractéristiques.

Il est effectivement courant, lors de l'emploi des caractéristiques d'Haralick, de n'utiliser qu'un sous ensemble de celles-ci. Pour ce faire, nous avons établi les corrélations existant entre les caractéristiques extraites des images d'Ubiris en effectuant une analyse par composantes principales. Les résultats de cette analyse sont résumés dans la figure 9.5, montrant le cercle des corrélations entre les 11 premières caractéristiques.

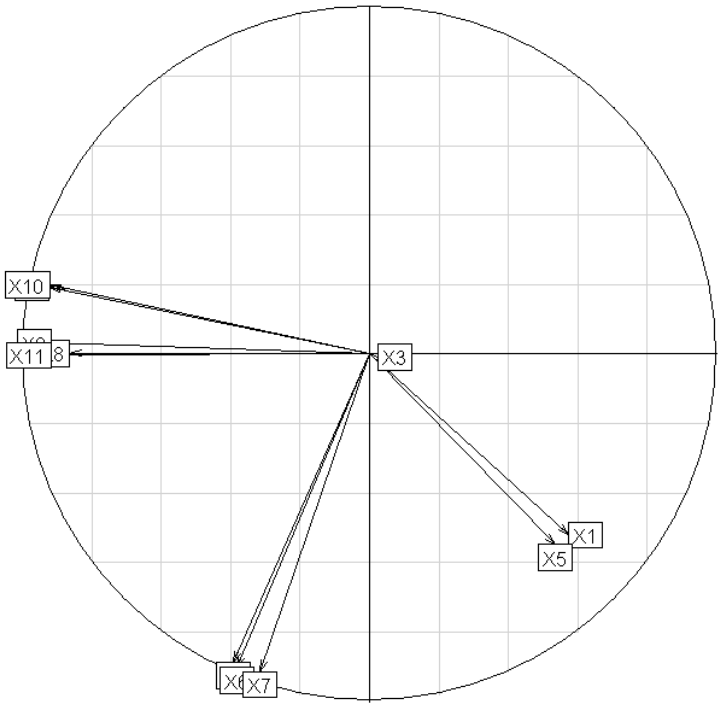


Figure 9.5 – Cercle des corrélations entre les caractéristiques d'Haralick.

Le cercle se lit de la manière suivante :
- deux variables proches sur le cercle sont corrélées entre elles ;
- deux variables situées de part et d'autre du cercle sont corrélées négativement ;
- deux variables situées à 90° l'une de l'autre sont indépendantes.

Le cercle nous permet de dégager les groupements de caractéristiques suivants : {3}, {8,9,11} (attendu étant donné les formulations de ces caractéristiques) {2,11}, {4,6,7} et {1,5}. Le singleton {3} n'apporte que peu d'informations et est donc écarté. Les groupes {2,10} et {8,9,11} sont eux-mêmes très corrélés.

Nous avons choisi d'employer les caractéristiques suivantes :
– le moment des différences inverses f_5 ;
– le moyenne des sommes f_6 ;
– l'entropie des différences f_{11}.
Le réseau NB_s résultant sera donc constitué de 4 variables : 3 caractéristiques et la classe des pixels.

9.7 Implémentation

La classification est effectuée sur une base d'images constituées de 50 images issues de la base Ubiris. Chaque image est issue d'un sujet différent afin d'éviter toute corrélation fortuite entre images d'un même individu.

La base d'apprentissage est, elle, constituée de 15 images issues elles aussi de sujets différents entre eux ainsi que de ceux constituant la base de test. Si le nombre d'images paraît limité, il faut souligner que l'apprentissage s'effectue au niveau des pixels. Les images étant de dimensions 200×150 pixels, nous avons à notre disposition une base de 430 000 points (les bords de l'image ne sont pas traités du fait du traitement par les matrices de cooccurrence).

L'apprentissage est effectué de manière supervisée, à partir d'une segmentation manuelle de la base d'apprentissage constituant alors la vérité terrain.

Nous effectuons nos calculs sur les onze premières caractéristiques de texture (les formules de ces caractéristiques sont précisées en annexe B). Les autres caractéristiques n'ayant montré qu'un intérêt faible par rapport aux calculs supplémentaires qu'elles requéraient. De plus, nous réduisons le nombre de niveaux de gris de l'image à traiter à 16. Cette réduction permet de diminuer considérablement le nombre de calculs nécessaires sans pour autant causer une perte d'information notable concernant les textures présentes.

En tout, le modèle bayésien NB_s est par conséquent constitué de 11 variables caractéristiques et de la variable binaire correspondant à la classe. Afin de pouvoir traiter les valeurs continues des caractéristiques d'Haralick, nous avons employé une méthode de discrétisation fondée sur l'emploi du critère AIC, tirée de [Colot et al., 1994] et disponible via la toolbox *Structure Learning Package*.

Pour la classification (pas dans le cas de l'apprentissage), une pré-segmentation est opérée afin d'accélérer les traitements. Cette pré-segmentation est fondée sur le même principe que les méthodes exposées en section 9.4.2 : une transformée de Hough circulaire est appliquée à l'image après détection des contours. Ce système ne rencontre de problèmes, dans le cas d'images prises en conditions réelles, que lors de la segmentation entre pupille et iris. Nous l'employons ici afin de délimiter une zone carrée, de dimension égale au rayon extérieur de l'iris détecté.

La classification se déroule comme suit :

1. la pré-segmentation est effectuée et permet de réduire la zone de l'image à traiter ;
2. une fenêtre glissante, de dimension $f \times f$ pixels est appliquée à chaque pixel, successivement dans l'image ;

3. pour chaque pixel, les caractéristiques d'Haralick sont extraites et constituent une base de cas ;
4. la base précédemment extraite est fournie en entrée au modèle appris à partir de la base de 15 images ;
5. la classe d'appartenance de chaque pixel est inférée à partir des caractéristiques extraites ;
6. l'image ainsi obtenue est soumise à des opérateurs de morphologie mathématique afin d'éliminer les éventuels artefacts présents.

Les opérateurs morphologiques sont appliqués à l'image segmentée à l'aide d'un élément structurant SE, de forme circulaire et de rayon 2.

Les opérateurs employés sont l'érosion ($Ero_{SE}(I)$) et la dilatation ($Dil_{SE}(I)$) :

$$Ero_{SE}(I) = \{(x,y)|SE_{(x,y)} \subseteq I\} \tag{9.2}$$
$$Dil_{SE}(I) = \{(x,y)|SE_{(x,y)} \cap I \neq \emptyset\} \tag{9.3}$$

où (x, y) désigne les coordonnées du point de l'image I où est appliqué l'opérateur. Plus exactement, nous appliquons une ouverture suivie d'une fermeture (*i.e.* la suite d'opérations *érosion*, *dilatation*, *dilatation*, *érosion*,).

Préalablement aux expérimentations, il nous a fallu fixer la valeur de deux paramètres :
– la taille f de la fenêtre glissante ;
– la distance d employée dans le calcul des caractéristiques à partir de la matrice de cooccurrence des niveaux de gris (cf section B.1.1).

Une première série d'essais a été effectuée à partir du modèle bayésien *NB* afin de choisir ces paramètres. Par la suite, nous avons utilisé une dimension f, pour la fenêtre glissante, égale à 7 pixels ainsi qu'une distance du vecteur de déplacement égale à 2 pixels.

Enfin, la matrice d'occurrence, base du calcul des caractéristiques de texture, a été calculée sur quatre directions (0°, 45°, 90° et 135°), symétrisée puis moyennée afin de rendre l'ensemble invariant à la rotation.

9.7.1 Résultats

Nous avons évalué quantitativement les résultats des segmentations effectuées à partir des deux modèles *NB* et NB_s. Ces résultats sont regroupés sous la forme des matrices de confusion de la figure 9.6.

Afin de clarifier les résultats, nous les représentons sous formes de taux, dans la figure 9.7

Les résultats montrent que les modèles sont capables, le plus souvent, de reconnaître les parties de l'image n'appartenant pas à l'iris, avec un taux de faux négatif bas (0,65% et 3,2%). Ce résultat extrêmement bas, pour le modèle simplifié, est malheureusement contrebalancé par une faible capacité à reconnaître les parties de l'image appartenant effectivement à l'iris.

En regardant le détail des résultats pour chaque image, nous nous sommes aperçus que les modèles ont eu plus particulièrement du mal à identifier les iris clairs et/ou fortement texturés. Un exemple d'iris posant problème ainsi que le résultat de sa segmentation à l'aide du réseau bayésien naïf doté de onze caractéristiques sont donnés dans la figure 9.8. On voit ici que seuls

		Segmentation	
		Iris	$\overline{Iris}$
Image	*Iris*	194577	29997
	$\overline{Iris}$	26281	814945

(a) *NB*

		Segmentation	
		Iris	$\overline{Iris}$
Image	*Iris*	146703	77871
	$\overline{Iris}$	5466	835760

(b) NB-s

Figure 9.6 – Évaluations quantitatives de la segmentation à travers les matrices de confusion calculées sur l'ensemble des images de tests, suivant le modèle bayésien employé.

		Segmentation	
		Iris	$\overline{Iris}$
Image	*Iris*	86,6%	13,4%
	$\overline{Iris}$	3,2%	96,8%

(a) NB

		Segmentation	
		Iris	$\overline{Iris}$
Image	*Iris*	65,3%	34,7%
	$\overline{Iris}$	0,6%	99,4%

(b) NB-s

Figure 9.7 – Évaluations quantitatives de la segmentation à travers les matrices de confusion calculées sur l'ensemble des images de tests, suivant le modèle bayésien employé. Les valeurs représentées ici sont les différents taux.

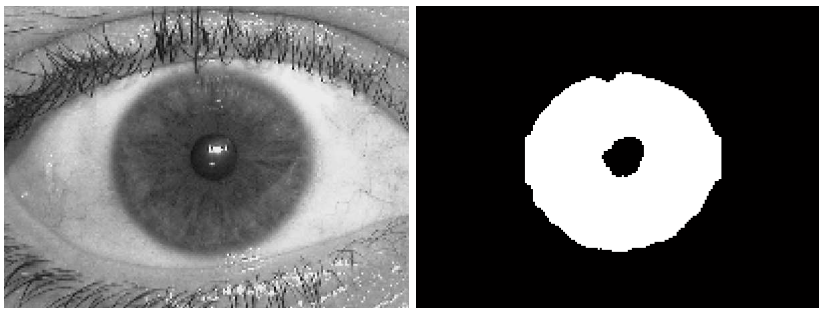
quelques pixels (68, pour être exact) ont été identifiés comme appartenant à l'iris. À l'opposé, à partir du même modèle que pour l'image précédente, les figures 9.9(a) et (b) montrent côte à côte l'image d'un autre iris segmenté et la segmentation obtenue à partir de cette image. Ici, une grande majorité des pixels de l'iris ont pu être identifiés.

(a) Image de la base Ubiris. (b) Segmentation par un modèle naïf.

Figure 9.8 – Image d'un iris et résultat de sa segmentation par le biais d'un réseau bayésien naïf à onze caractéristiques. Seuls quelques pixels ont pu être identifiés comme appartenant à l'iris.

Dans un premier temps, nous avons soupçonné une incomplétude ou un manque de diversité dans la base d'apprentissage d'être la cause d'un tel phénomène. Mais l'ajout d'images d'iris présentant des textures similaires n'eut, dans des essais ultérieurs, aucun effet.

La réduction du nombre de caractéristiques modélisées au sein du modèle naïf a eu pour effet



(a) Image de la base Ubiris. (b) Segmentation par un modèle naïf.

Figure 9.9 – Image d'un iris et résultat de sa segmentation par le biais d'un réseau bayésien naïf à onze caractéristiques. De nombreux pixels appartenant à l'iris ont pu être identifiés.

une dégradation des performances du modèle. Même si les variables font parties de groupes fortement corrélés, l'information apportée par une seule caractéristique de chaque groupe est clairement insuffisante pour permettre une segmentation efficace des pixels de l'iris.

Les temps de calcul demeurent de plus assez lourds du fait de l'emploi d'une classification par pixel. Il est nécessaire de parcourir l'ensemble de l'image afin d'extraire les différentes caractéristiques et le temps mis est de près de 110 secondes dans le cas du modèle réduit NB_s et de 200 pour le modèle NB, même à l'aide de la pré-segmentation du cercle extérieur de l'iris.

9.8 Conclusion

Les résultats montrent que le modèle bayésien est capable d'effectuer une tâche de segmentation efficace.

Certaines images, cependant, semblent demeurer réfractaires à la segmentation par un modèle bayésien.

Une explication à ce phénomène est très certainement l'erreur humaine. Il n'existe pas en effet de vérité terrain "officielle" pour la base Ubiris ; la conséquence étant que nous avons nous-mêmes produit notre propre vérité terrain. Or, il est certainement de nombreux défauts dans l'image (perte de focus, reflet insensible sur l'iris) perturbant les valeurs des caractéristiques que nous n'avons pas, nous-mêmes, pu établir lors de la création de nos bases d'apprentissage.

Si le travail réalisé ici avait avant tout pour volonté de démontrer les capacités des réseaux bayésiens en tant que classificateurs sur un terrain inhabituel et non de résoudre la problématique de la segmentation de l'iris, ce dernier enjeu pourrait certainement être entrepris. Certains apports à cette méthode pouvant, entre autres, être l'exploitation de techniques pouvant permettre l'accélération du traitement et l'amélioration de la segmentation.

D'autre part, nous n'avons présenté dans ce chapitre que les résultats liés à un seul type de classificateur bayésien : le modèle naïf. Nous avons essayé, lors du développement du projet, d'employer des modèles plus fortement connectés tels que les réseaux bayésiens naïfs augmentés, les réseaux bayésiens ou encore les arbres retournés par la méthode MWST. Les

résultats, surprenants, nous paraissent aujourd'hui encore difficilement explicables. L'ajout de connexions au classificateur eut en effet pour résultat des modèles classifiant systématiquement la quasi totalité des points de l'image dans la classe $\overline{Iris}$. L'analyse des structures (répartition des caractéristiques au sein des différents modèles par rapport à leur corrélation, définie par l'ACP) ne nous a pas permis d'obtenir une réelle explication à ce phénomène.

Quatrième partie

Conclusions et perspectives

Chapitre 10

Conclusion

Ce travail de thèse nous a permis d'établir quelques uns des principaux tenants et aboutissants de l'approche évolutionnaire de l'apprentissage de la structure d'un réseau bayésien.

Dans un premier temps, nous avons pu étudier la problématique de l'apprentissage de structures à travers les principales notions et définitions ainsi qu'à travers un panorama des méthodes existantes.

Les méthodes évolutionnaires ont ensuite été définies puis étudiées afin de pouvoir les employer de manière adaptée pour l'apprentissage de structures. Nous avons par la suite proposé un algorithme génétique permettant un tel apprentissage, suivant une procédure de recherche et d'évaluation dans l'espace des structures avant d'augmenter cet algorithme par diverses approches issues des travaux combinés des domaines de l'algorithmique évolutionnaire et des modèles bayésiens.

L'exploitation des propriétés de l'espace des équivalents de Markov a été une première étape. En combinant la non-redondance proposée par les graphes représentant des classes d'équivalence à une méthode de *niching* séquentiel, nous avons pu, de manière simple, améliorer les performances de l'algorithme génétique. Cette méthode revenant à pénaliser l'évaluation de certaines solutions par le biais de leurs classes d'équivalences.

Dans la lignée de certains travaux récents visant à hybrider certaines techniques d'exploration de l'espace des solutions, nous avons procédé à une combinaison de la méthode précédente avec un schéma de répartition des individus de la population dans l'espace. Cette dernière méthode s'est montrée généralement plus performante, parvenant à au moins égaler les performances de l'algorithme glouton sur l'espace des équivalents, GES.

Nous avons aussi exploité la piste de l'adaptativité des opérateurs, thématique très documentée, en proposant une méthode d'adaptativité de l'opérateur de mutation. À la manière d'un processus de mémorisation des réussites et erreurs passées venant renforcer le processus évolutionnaire, les conclusions quant à cette dernière stratégie s'avèrent mitigés. La complexité même du problème de l'apprentissage de structures ainsi qu'une représentation *a priori* inadaptée des solutions dans notre méthode figurent parmi les raisons derrière les performances moindres de cette stratégie adaptative.

Le premier résultat de notre travail est que les méthodes évolutionnaires peuvent être une approche judicieuse dans les cas vraisemblables où l'on cherche à établir la structure d'un réseau bayésien à partir d'une base de cas de taille limitée. En effet, dans ce cas précis, la multiplicité des optima locaux ainsi que l'absence d'*a priori* sur la modélisation viennent fréquemment entraver le fonctionnement d'algorithmes tels que les algorithmes gloutons.

De plus, nos propositions de méthodes de *niching*, hybridées ou non, se sont avérées efficaces en permettant de recouvrer des structures non seulement plus vraisemblables que celles retournées par les méthodes existantes, mais aussi proches du modèle sous-jacent aux données considérées.

Une autre partie de nos travaux a consisté à évaluer le potentiel des réseaux bayésiens en tant que classifieurs apppliqués à l'image et ce, dans le cadre d'une problématique précise : la segmentation de la zone de l'iris dans l'image d'un œil humain. Le modèle naïf utilisé a permis d'obtenir des taux de segmentation très performants, de l'ordre de 90%, en moyenne. Si le système développé reste avant tout une étude théorique, les résultats nous permettent d'envisager l'exploitation future des modèles bayésiens dans le domaine de la reconnaissance de formes bien qu'il nous semble impératif de combiner, du fait de la complexité de la tâche, les modèles probabilistes aux connaissances et outils d'ores et déjà employés pour le traitement de l'image : croissance de région, pré-segmentation, approche multi-résolution, etc.

Chapitre 11

Perspectives

Le travail mené dans cette thèse a permis de pouvoir confronter les caractéristiques d'une problématique d'intérêt – l'apprentissage de la structure d'un réseau bayésien – à celles d'un ensemble de méthodes tout aussi populaire – les méthodes évolutionnaires –. S'il est naturel de penser que ces deux domaines devaient un jour faire l'objet d'une étude commune, ce qui a d'ores et déjà été le cas (cf section 5.5), on peut être surpris du fait que la plupart de ces méthodes ont principalement voulu adapter le problème au solveur et non l'inverse. Or les limitations en performances de ces approches sont, à notre avis, pour la plupart inhérentes à l'approche du problème et non nécessairement à celui-ci même.

Si l'on s'intéresse en particulier au cas des algorithmes employant une stratégie de *niching*, combinée ou non à un schéma de répartition de la population, ces stratégies pourraient être améliorées par l'emploi d'une distance définie directement sur l'espace des graphes essentiels, servant eux-mêmes dans nos travaux à la définition de niches. Des travaux récents [Tsamardinos et al., 2006] ont ainsi déterminé une distance d'édition au sein de l'espace de graphes essentielles : la distance de Hamming structurelle. Les résultats de l'emploi de cette distance en conjugaison avec une approche de *niching* spatial classique – dont l'emploi était dans notre cas contraint par la définition d'une distance *ad hoc* dans l'espace de recherche – serait un sujet d'intérêt, dans la continuité de nos travaux.

L'un des principaux désavantages des méthodes évolutionnaires appliquées à l'apprentissage de structure, outre le temps de calcul, est d'optimiser la population en cours en fonction de la seule *fitness* alors couramment égale à une fonction d'évaluation de structures telle que le critère BIC. Bien que justifiée, cette approche entraîne, dans le cas de l'apprentissage de structures, un certain nombre de problèmes intimement liés aux défauts des méthodes d'évaluation issue du domaine des réseaux bayésiens. Il n'existe en effet pas, à l'heure actuelle, de méthode d'évaluation parfaitement fiable pour une structure, preuve s'il en est, le nombre important de mesures employées : AIC, BIC, MDL, BDeu. La fiabilité des tests d'indépendance statistique est aussi un problème, si l'on souhaite aborder le problème par cette voie.

Les scores aussi bien que les méthodes basées sur la détection d'indépendances probabilistes rencontrent toutes des problèmes dans le cas de bases d'apprentissage de tailles restreintes. Une possibilité, dès lors, serait de pouvoir combiner différentes mesures afin de réussir à atteindre un compromis entre vraisemblance structurelle, du point vue de la mesure mathématique, et de

la vraisemblance définie par un expert. Or, l'optimisation de problèmes à contraintes multiples est justement une des pistes étudiées par l'algorithmique évolutionnaire [Deb, 2001]. On peut penser qu'il serait possible, par le biais d'un algorithme évolutionnaire *ad hoc* de parvenir à l'obtention d'une solution "raisonnable" vis-à-vis de divers critères (règles graphiques, contraintes de succession ou d'ascendance locales à certaines variables, etc).

Si la distributivité des calculs nous laisse espérer une amélioration sensible des temps de calculs liés à l'apprentissage de modèles complexes, la considération simultanée de plusieurs critères, aussi bien théoriques que pratiques, pourrait amener la solution évolutionnaire à pouvoir produire rapidement une "bonne" solution à partir de données peu nombreuses.

Les domaines des modèles bayésiens et des méthodes évolutionnaires sont riches, respectivement, en problématiques et en solutions. Certains pans de la problématique de l'apprentissage de la structure d'un réseau bayésien n'ont pas été traités dans ce travail de thèse. Ainsi la problématique des données manquantes est une question faisant l'objet de nombreux travaux (cf section 4.1.2). Même si, nous l'avions évoqué, les méthodes évolutionnaires ont déjà fait l'objet d'une application dans ce domaine [Myers et al., 1999], celle-ci a ignoré les nombreux cas de figure pouvant se produire dans cette seule problématique [François, 2006].

Enfin, dans le domaine plus général de l'apprentissage de structures par un algorithme évolutionnaire, plusieurs avancées ont été faites en parallèle à ce travail de thèse ; ces avancées ont consisté en une simplification des représentations usuellement employées en limitant la recherche à un squelette établi par une série de tests statistiques tandis que des travaux prometteurs se sont employés à déterminer une structure adéquate à partir d'une recherche sur l'ensemble des ordres topologiques envisageables. Une piste intéressante pourrait alors consister à combiner ces deux approches en permettant une recherche sur l'espace des ordres topologiques (d'ores et déjà limité par rapport à celui des structures) en tenant compte du squelette préétabli.

Bibliographie

[Acid et de Campos, 1996] Acid, S. et de Campos, L. (1996). A hybrid methodology for learning belief networks : Benedict. *International Journal of Approximate Reasoning*, 27 :235–262.

[Acid et de Campos, 2003] Acid, S. et de Campos, L. M. (2003). Searching for bayesian network structures in the space of restricted acyclic partially directed graphs. *Journal of Artificial Intelligence Research*, 18 :445–490.

[Akaike, 1970] Akaike, H. (1970). Statistical predictor identification. *Annals of the Institute of Statistical Mathematics*, 22 :203–217.

[Allanach et al., 2004] Allanach, J., Tu, H., Singh, S., Pattipati, K., et Willett, P. (2004). Detecting, tracking and counteracting terrorist networks via hidden markov models. In *IEEE Aerospace Conference*.

[Anastasoff, 1999] Anastasoff, S. J. (1999). Evolving mutation rates for the self-optimisation of genetic algorithms. In *Advances in Artificial Life, ECAL'99 : 5th European Conference on Artificial Life*, pages 133–139, Lausanne, Switzerland. Springer.

[Andersson et al., 1995] Andersson, S., Madigan, D., et Perlman, M. (1995). A characterization of markov equivalence classes for acyclic digraphs. Technical Report 287, Department of Statistics, University of Washington.

[Angeline, 1995] Angeline, P. J. (1995). Adaptive and self-adaptive evolutionary computations. In Palaniswami, M. et Attikiouzel, Y., editors, *Computational Intelligence : A Dynamic Systems Perspective*, pages 152–163. IEEE Press.

[Antonisse, 1989] Antonisse, J. (1989). A new interpretation of schema notation that overturns the binary encoding constraint. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 86–97, San Francisco. Morgan Kaufmann.

[Auger, 2004] Auger, A. (2004). *Contributions théoriques et numériques à l'optimisation continue par Algorithmes Evolutionnaires*. PhD thesis, Université Paris 6.

[Baluja, 1994] Baluja, S. (1994). Population-based incremental learning : A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.

[Bäck, 1992] Bäck, T. (1992). Self-adaptation in genetic algorithms. In Varela, F. J. et Bourgine, P., editors, *Proceedings of the First European Conference on Artificial Life*, pages 227–235, Cambridge, MA. MIT Press.

[Bäck, 1993] Bäck, T. (1993). Optimal mutation rates in genetic search. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 2–8.

[Bäck et al., 2000] Bäck, T., Eiben, A. E., et van der Vaart, N. A. L. (2000). An empirical study on gas without parameters. In *PPSN VI : Proceedings of the 6th International Conference on Parallel Problem Solving from Nature, Paris, France, September 18-20*, pages 315–324.

[Bäck et Schütz, 1996] Bäck, T. et Schütz, M. (1996). Intelligent mutation rate control in canonical genetic algorithms. In *Proceedings of the International Symposium on Methodologies for Intelligent Systems.*, pages 158–167.

[Beal, 2003] Beal, M. (2003). *Variational Algorithms for Approximate Bayesian Inference.* PhD thesis, Gatsby Computational Neuroscience Unit, University College, London.

[Beasley et al., 1993] Beasley, D., Bull, D. R., et Martin, R. R. (1993). A sequential niche technique for multimodal function optimization. *Evolutionary Computation*, 1(2) :101–125.

[Beinlich et al., 1989] Beinlich, I. A., Suermondt, H. J., Chavez, R. M., et Cooper, G. F. (1989). The alarm monitoring system : A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, pages 247–256, London, England.

[Binder et al., 1997] Binder, J., Koller, D., Russell, S. J., et Kanazawa, K. (1997). Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2-3) :213–244.

[Blanco et al., 2003] Blanco, R., Inza, I., et Larrañaga, P. (2003). Learning bayesian networks in the space of structures by estimation of distribution algorithms. *International Journal of Intelligent Systems*, 18(2) :205–220.

[Bonet et al., 1996] Bonet, J. S. D., Isbell, C. L., et Viola, P. (1996). Mimic : Finding optima by estimating probability densities. In *Proceedings of Neural Information Processing Systems*, pages 424–430.

[Bouckaert, 1994] Bouckaert, R. (1994). Properties of bayesian belief network learning algorithms. In *Proceedings of the 10th Annual Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 102–10, San Francisco, CA. Morgan Kaufmann.

[Bouckaert, 1993] Bouckaert, R. R. (1993). Probabilistic network construction using the minimum description length principle. *Lecture Notes in Computer Science*, 747 :41–48.

[Bozdogan, 1987] Bozdogan, H. (1987). Model selection and akaike's information criteria (AIC) : The general theory and its analytical extentions. *Psychometrika*, 52 :354–370.

[Buntine, 1991] Buntine, W. (1991). Theory refinement of bayesian networks. In *In Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 52–60. American Mathematical Society.

[Canny, 1986] Canny, F. J. (1986). A computational approach to edge detection. *Transactions on Pattern Analysis and Machine Intelligence*, 8(6) :679–698.

[Cantu-Paz, 1997] Cantu-Paz, E. (1997). A survey of parallel genetic algorithms. Technical Report 97003, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign.

[Charniak, 1991] Charniak, E. (1991). Bayesian networks without tears. *AI Magazine*, 12(4) :50–63.

[Cheeseman et al., 1988] Cheeseman, P., Self, M., Kelly, J., Taylor, W., Freeman, D., et Stutz, J. (1988). Bayesian classification. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 607–617, St. Paul, MN.

[Chellapilla et Fogel, 1999] Chellapilla, K. et Fogel, D. (1999). Fitness distributions in evolutionary computation : motivation and examples in the continuous domain. *BioSystems*, 54 (1-2) :15–29.

[Cheng et al., 2002] Cheng, J., Bell, D. A., et Liu, W. (2002). Learning belief networks from data : An information theory based approach. *Artificial Intelligence*, 1-2 :43–90.

[Chickering, 1995] Chickering, D. (1995). A transformational characterization of bayesian network structures. In Hanks, S. et Besnard, P., editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 87–98.

[Chickering, 2002a] Chickering, D. (2002a). Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3 :507–554.

[Chickering et al., 1995] Chickering, D., Geiger, D., et Heckerman, D. (1995). Learning bayesian networks : Search methods and experimental results. In *Proceedings of the fifth Conference on Artificial Intelligence and Statistics*, pages 112–128.

[Chickering, 1996] Chickering, D. M. (1996). Learning equivalence classes of bayesian network structures. In Horvitz, E. et Jensen, F. V., editors, *Proceedings of the 12th Annual Conference on Uncertainty in Artificial Intelligence*, pages 150–157. Morgan Kaufmann.

[Chickering, 2002b] Chickering, D. M. (2002b). Learning equivalence classes of bayesian-network structures. *Journal of Machine Learning Research*, 2 :445–498.

[Chickering et al., 1994] Chickering, D. M., Geiger, D., et Heckerman, D. (1994). Learning bayesian networks is NP-hard. Technical report, Microsoft Research.

[Chickering et Meek, 2003] Chickering, D. M. et Meek, C. (2003). Monotone DAG faithfulness : A bad assumption. Technical Report MSR-TR-2003-16, Microsoft Research.

[Chickering et al., 2003] Chickering, D. M., Meek, C., et Heckerman, D. (2003). Large-sample learning of bayesian networks is hard. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence,* Acapulco, Mexico, pages 162–169. Morgan Kaufmann.

[Chow et Liu, 1968] Chow et Liu (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3) :462–467.

[Cobb et Shenoy, 2006] Cobb, B. R. et Shenoy, P. P. (2006). Inference in hybrid bayesian networks with mixtures of truncated exponentials. *International Jounal of Approximate Reasoning*, 41(3) :257–286.

[Cohoon et al., 1987] Cohoon, J. P., Hedge, S. U., Martin, W. N., et Richards., D. (1987). Punctuated equilibria : A parallel genetic algorithm. In Grefenstette, J. J., editor, *Genetic algorithms and their applications : Proceedings of the second International Conference on Genetic Algorithms*, pages 148–154. Lawrence Erlbaum Associates.

[Colot et al., 1994] Colot, O., Olivier, C., Courtellemont, P., et El Matouat, A. (1994). Information criteria and abrupt changes in probability laws. In *Signal Processing VII : Theories and Applications*, pages 1855–1858.

[Cooper et Herskovits, 1992] Cooper, G. et Herskovits, E. (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9 :309–347.

[Cooper, 1987] Cooper, G. F. (1987). Probabilistic inference using belief networks is NP-hard. Technical Report KSL-87-27, Medical Computer Science Group, Stanford University, Stanford, CA.

[Cormen et al., 1994] Cormen, T., Leiserson, C., et Rivest, R. (1994). *Introduction à l'algorithmique*. Dunod.

[Cotta et Muruzábal, 2002] Cotta, C. et Muruzábal, J. (2002). Towards a more efficient evolutionary induction of bayesian networks. In *PPSN VII : Parallel Problem Solving from Nature, 7th International Conference, Granada, Spain, September 7-11*, pages 730–739.

[Dagum et Luby, 1993] Dagum, P. et Luby, M. (1993). Approximate probabilistic reasoning in bayesian belief networks is NP-hard. *Artificial Intelligence*, 60 :141–153.

[D'Ambrosio, 1993] D'Ambrosio, B. (1993). Incremental probabilistic inference. In *UAI '93 : Proceedings of the Ninth Annual Conference on Uncertainty in Artificial Intelligence*, pages 301–308, Providence, Washington, DC, USA. The Catholic University of America.

[Dash et Druzdzel, 1999] Dash, D. et Druzdzel, M. J. (1999). A hybrid anytime algorithm for the construction of causal models from sparse data. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 142–149.

[Daugman, 1993] Daugman, J. (1993). High confidence visual recognition of persons by a test of statistical independence. *Image Pattern Analysis and Machine Intelligence*, 15(11) :1148–1161.

[Daugman, 2007] Daugman, J. G. (2007). New methods in iris recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 37(5) :1167–1175.

[Davis, 1991] Davis, L. (1991). *Handbook of Genetic Algorithms.* van Nostrand Reinhold, New York.

[de Campos et Castellano, 2007] de Campos, L. M. et Castellano, J. G. (2007). Bayesian network learning algorithms using structural restrictions. *International Journal of Approximate Reasoning*, 45(2) :233–254.

[De Jong, 1992] De Jong, K. (1992). Are genetic algorithms function optimizers ? In *Parallel Problem Solving from Nature 2, PPSN-II, Brussels*, pages 3–14.

[De Jong, 2001] De Jong, K. (2001). *Evolutionary Computation : A Unified Approach*. MIT Press.

[De Jong, 1975] De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems*. PhD thesis, University of Michigan.

[Deb, 2001] Deb, K. (2001). *Multi-objective optimization using genetic algorithms*. Wiley.

[Deb et Agrawal, 1995] Deb, K. et Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex Systems*, 9 :115–148.

[Dechter, 1997] Dechter, R. (1997). Mini-buckets : a general scheme for approximation in automated reasoning. In *Proceedings of the fifteenth International Joint Conference on Artifical Intelligence (IJCAI)*, pages 1297–1302.

[Delaplace et al., 2006] Delaplace, A., Brouard, T., et Cardot, H. (2006). Two evolutionary methods for learning bayesian network structures. In *Proceedings of the 2006 International Conference on Computational Intelligence and Security (CIS 2006)*, volume 4456 of *Lecture Notes in Artificial Intelligence*, pages 73–80, Guangzhou, China. Springer.

[Delaplace et al., 2007a] Delaplace, A., Brouard, T., et Cardot, H. (2007a). Apprentissage de la structure d'un réseau bayésien par un algorithme génétique. *Revue d'Intelligence Artificielle*, 21(3) :333–352.

[Delaplace et al., 2007b] Delaplace, A., Brouard, T., et Cardot, H. (2007b). Détermination évolutionnaire de classes d'équivalences de structures de réseaux bayésiens. In *Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision FRANCORO V / ROADEF 2007*, Grenoble, France. Presses Universitaires de Grenoble.

[Dempster et al., 1977] Dempster, A., Laird, N., et Rubin, D. (1977). Maximum likelihood for incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B 39 :1–38.

[Domingos et Pazzani, 1996] Domingos, P. et Pazzani, M. J. (1996). Beyond independence : Conditions for the optimality of the simple bayesian classifier. In *Machine Learning, Proceedings of the Thirteenth International Conference (ICML '96)*, pages 105–112, Bari, Italy. Morgan Kaufmann.

[Dor et Tarsi, 1992] Dor, D. et Tarsi, M. (1992). A simple algorithm to construct a consistent extension of a partially oriented graph. Technical Report Technical Report R-185, Cognitive Systems Laboratory, UCLA Computer Science Department.

[Draper et Hanks, 1994] Draper, D. et Hanks, S. (1994). Localized partial evaluation of belief networks. In *Proceedings of the 10th Annual Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 170–17, San Francisco, CA. Morgan Kaufmann.

[Droste et al., 2001] Droste, S., Jansen, T., et Wegener, I. (2001). Dynamic parameter control in simple evolutionary algorithms. In Martin, W. N. et Speards, W. M., editors, *Proceedings of the Sixth Workshop on the Foundations of Genetic Algorithms*, pages 275–294, San Francisco CA.

[Eaton, 2007] Eaton, D. (2007). Bayesian network structure learning for the uncertain experimentalist. Master's thesis, University of British Columbia.

[Eiben et al., 1999] Eiben, A. E., Hinterding, R., et Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2) :124–141.

[Eiben et al., 2004] Eiben, A. E., Marchiori, E., et Valkó, V. A. (2004). Evolutionary algorithms with on-the-fly population size adjustment. In *PPSN VIII : Proceedings of the 8th International Conference on Parallel Problem Solving from Nature*, pages 41–50.

[Eiben et al., 2006] Eiben, A. E., Schut, M. C., et de Wilde, A. R. (2006). Is self-adaptation of selection pressure and population size possible ? - a case study. In *PPSN IX : Proceedings of the 9th International Conference on Parallel Problem Solving from Nature*, pages 900–909.

[Eiben et Smith, 2003] Eiben, A. E. et Smith, J. E. (2003). *Introduction to Evolutionary Computing*. Springer.

[El-Yaniv et al., 1997] El-Yaniv, R., Fine, S., et Tishby, N. (1997). Agnostic classification of markovian sequences. In *Advances in Neural Information Processing Systems 10, NIPS Conference, Denver, Colorado, USA*.

[Eldredge et Gould, 1972] Eldredge, N. et Gould, S. (1972). Punctuated equilibria : an alternative to phyletic gradualism. In *Models of Paleobiology*, pages 82–115. Freeman Cooper and co, San Francisco CA.

[Elidan, 2004] Elidan, G. (2004). *Learning Hidden Variables in Probabilistic Graphical Models*. PhD thesis, Hebrew University.

[Elidan et Friedman, 2001] Elidan, G. et Friedman, N. (2001). Learning the dimensionality of hidden variables. In *UAI '01 : Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, University of Washington, Seattle, Washington, USA, August 2-5, 2001*, pages 144–151.

[Etxeberria et al., 1997] Etxeberria, R., Larrañaga, P., et Picaza, J. M. (1997). Analysis of the behaviour of genetic algorithms when learning bayesian network structure from data. *Pattern Recognition Letters*, 18(11-13) :1269–1273.

[Fennell et Wishner, 1998] Fennell, M. T. et Wishner, R. P. (1998). Battlefield awareness via synergistic SAR and MTI exploitation. *IEEE Aerospace and Electronic Systems Magazine*, 13(2) :39–43.

[Fogel et al., 1966] Fogel, L. J., Owens, A. J., et Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons, New York.

[Forrest, 1985] Forrest, S. (1985). Documentation for prisoners dilemma and norms programs that use the genetic algorithm. University of Michigan, Ann Arbor, MI.

[Francois et Leray, 2004] Francois, O. et Leray, P. (2004). BNT structure learning package : Documentation and experiments. Technical report, Laboratoire PSI.

[François, 2006] François, O. (2006). *De l'identification de structure de réseaux bayésiens à la reconnaissance de formes à partir d'informations complètes ou incomplètes.* PhD thesis, Institut national des sciences appliquées de Rouen.

[François et Leray, 2004] François, O. et Leray, P. (2004). étude comparative d'algorithmes d'apprentissage de structure dans les réseaux bayésiens. *Journal Electronique d'Intelligence Artificielle*, 5(39) :1–19.

[Friedman, 1997] Friedman, N. (1997). Learning bayesian networks in the presence of missing values and hidden variables. In *Proceedings of the 14th International Conference on Machine Learning*, pages 125–133. Morgan Kaufmann.

[Friedman, 1998] Friedman, N. (1998). The bayesian structural EM algorithm. In *Fourteenth Conf. on Uncertainty in Artificial Intelligence (UAI)*, pages 129–138.

[Friedman et al., 1997] Friedman, N., Geiger, D., et Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29 :131–163.

[Friedman et Goldszmidt, 1996] Friedman, N. et Goldszmidt, M. (1996). Discretizing continuous attributes while learning bayesian networks. In *ICML*, pages 157–165.

[Friedman et Koller, 2000] Friedman, N. et Koller, D. (2000). Being bayesian about network structure. In *Proceedings of the 16th Annual Conference on Uncertainty in Artificial Intelligence*, pages 201–2, San Francisco, CA. Morgan Kaufmann.

[Fu, 2005] Fu, L. D. (2005). A comparison of state-of-the-art algorithms for learning bayesian network structure from continuous data. Master's thesis, Faculty of the Graduate School of Vanderbilt University.

[Geman et Geman, 1984] Geman, S. et Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6 :721–741.

[Gilks et al., 1996] Gilks, W., Richardson, S., et Spiegelhalter, D. (1996). *Markov Chain Monte carlo in Practice*. Chapman & Hall.

[Gillispie et Perlman, 2002] Gillispie, S. B. et Perlman, M. D. (2002). The size distribution for markov equivalence classes of acyclic digraph models. *Artificial Intelligence archive*, 141(1/2) :137–155.

[Glickman et Sycara, 2000] Glickman, M. et Sycara, K. (2000). Reasons for premature convergence of self-adapting mutation rates. In *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, pages 62 – 69.

[Goldberg, 1989] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional.

[Goldberg et Richardson, 1987] Goldberg, D. E. et Richardson, J. (1987). 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. Lawrence Erlbaum Associates, Inc.

[Gomez, 2004] Gomez, J. (2004). Self adaptation of operator rates in evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2004)*, pages 1162–1173.

[Grefenstette, 1981] Grefenstette, J. (1981). Parallel adaptive algorithms for function optimization. Technical Report CS-81-19, Computer Science Department, Carnegie Mellon University, Vanderbilt University, Nashville, TN.

[Haralick et al., 1973] Haralick, R., Shanmugan, K., et Distein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 3 :610–621.

[He et Shi, 2007] He, X. et Shi, P. (2007). A new segmentation approach for iris recognition based on hand-held capture device. *Pattern Recognition*, 40(4) :1326–1333.

[Heckerman, 1995] Heckerman, D. (1995). A tutorial on learning bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, Redmond, WA.

[Heckerman et Geiger, 1994] Heckerman, D. et Geiger, D. (1994). A characterization of the dirichlet distribution through global and local independence. *The Annals of Statistics*, 25(3) :1344–1369.

[Heckerman et al., 1995a] Heckerman, D., Geiger, D., et M.Chickering, D. (1995a). Learning bayesian networks : The combination of knowledge and statistical data. *Machine Learning*, 20 :197–243.

[Heckerman et al., 1995b] Heckerman, D., Mamdani, A., et Wellman, M. P. (1995b). Real world applications of bayesian networks. *Communications of the ACM*, 38(3) :24–30.

[Henrion, 1988] Henrion, M. (1988). Propagation of uncertainty by probabilistic logic sampling in bayes networks. *Uncertainty in Artificial Intelligence*, 2 :149–164.

[Holland, 1975] Holland, J. H. (1975). *Adaptation in natural and artificial systems.* The University of Michigan Press, Ann Arbor.

[Hu et Goodman, 2004] Hu, J. et Goodman, E. D. (2004). Robust and efficient genetic algorithms with hierarchical niching and a sustainable evolutionary computation model. In *Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004, Proceedings, Part I*, pages 1220–1232.

[Huang et Darwiche, 1996] Huang, C. et Darwiche, A. (1996). Inference in belief networks : A procedural guide. *International Journal of Approximate Reasoning*, 15 (3) :225–263.

[Hurvich et Tsai, 1989] Hurvich, C. M. et Tsai, C.-L. (1989). Regression and time series model selection in small samples. *Biometrika*, 76(2) :297–307.

[Jaakkola et Jordan, 1999] Jaakkola, T. et Jordan, M. I. (1999). Variational probabilistic inference and the qmr-dt network. *Journal of Artificial Intelligence Research*, 10 :291–322.

[Jaronski et al., 2001] Jaronski, W., Bloemer, J., Vanhoof, K., et Wets, G. (2001). Use of bayesian belief networks to help understand online audience. In *Proceedings of the Data Mining Marketing Applications Workshop ECML/PKDD*, Freiburg, Germany.

[Jensen, 1996] Jensen, F. (1996). *An Introduction to Bayesian Networks*. Springer Verlag, New York.

[Jensen et al., 1990] Jensen, F., Lauritzen, S., et Olesen, K. (1990). Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quaterly*, 4 :269–282.

[Jordan, 1998] Jordan, M. (1998). *Learning in Graphical Models.* Dordecht, The Netherlands : Kluwer Academic Publishers.

[Jordan, 2004] Jordan, M. I. (2004). Graphical models. *Statistical Science (Special Issue on Bayesian Statistics)*, 19(1) :140–155.

[Kallel et al., 2001] Kallel, L., Naudts, B., et Rogers, A., editors (2001). *Theoretical Aspects of Evolutionary Computing*. Springer, Berlin.

[Kayaalp et Cooper, 2002] Kayaalp, M. et Cooper, G. F. (2002). A bayesian network scoring metric that is based on globally uniform parameter priors. In *Proceedings of the Eighteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-2002)*, pages 251–258.

[Kjærulff, 1994] Kjærulff, U. (1994). Reduction of computational complexity in bayesian networks through removal of weak dependences. In *UAI '94 : Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 374–382.

[Kocka et al., 2001] Kocka, T., Bouckaert, R. R., et Studený, M. (2001). On characterizing inclusion of bayesian networks. In *UAI '01 : Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, University of Washington, Seattle, Washington, USA, August 2-5, 2001*, pages 261–268.

[Koehler, 1997] Koehler, G. J. (1997). New directions in genetic algorithm theory. *Annals of Operations Research*, 75 :49–68.

[Koza, 1989] Koza, J. R. (1989). Hierarchical genetic algorithms operating on populations of computer programs. In Sridharan, N. S., editor, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence IJCAI-89*, volume 1, pages 768–774. Morgan Kaufmann.

[Koza, 1992] Koza, J. R. (1992). *Genetic Programming : On the programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, Massachusetts.

[Krause, 1999] Krause, P. J. (1999). Learning probabilistic networks. *The Knowledge Engineering Review archive*, 13(4) :321–351.

[Kreinovich et al., 1993] Kreinovich, V., Quintana, C., et Fuentes, O. (1993). Genetic algorithms : What fitness scaling is optimal ? *Cybernetics and Systems*, 24(1) :9–26.

[Lacey et MacNamara, 2000] Lacey, G. et MacNamara, S. (2000). Context-aware shared control of a robot mobility aid for the elderly blind. *I. J. Robotic Res.*, 19(11) :1054–1065.

[Lam et Bacchus, 1994] Lam, W. et Bacchus, F. (1994). Learning bayesian belief networks : An approach based on the MDL principle. *Computational Intelligence*, 10 :269–294.

[Langley et al., 1992] Langley, P., Iba, W., et Thompson, K. (1992). An analysis of bayesian classifiers. In (Ed.), W. R. S., editor, *Proceedings of the 10th National Conference on Artificial Intelligence.*, pages 223–228, San Jose, CA,. The AAAI Press.

[Larrañaga et al., 2000] Larrañaga, P., Etxeberria, R., Lozano, J., et Peña, J. (2000). Combinatorial optimization by learning and simulation of bayesian networks. In *Proceedings of the Conference in Uncertainty in Artificial Intelligence, UAI 2000*, pages 343–352.

[Larrañaga et al., 1996] Larrañaga, P., Kuijpers, C., Murga, R., et Yurramendi, Y. (1996). Learning bayesian network structures by searching for the best ordering with genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 26(4) :487–493.

[Larrañaga et Lozano, 2001] Larrañaga, P. et Lozano, J. A. (2001). *Estimation of Distribution Algorithms : A New Tool for Evolutionary Computation (Genetic Algorithms and Evolutionary Computation)*. Kluwer Academic Press.

[Larrañaga et al., 2001] Larrañaga, P., Lozano, J. A., et Bengoetxea, E. (2001). Estimation of distribution algorithms based on multivariate normal and gaussian networks. Technical Report KZZA-1K-1-01, Department of Computer Science and Artificial Intelligence, University of the Basque Country, Donostia, Spain.

[Larranaga et al., 1996] Larranaga, P., Poza, M., Yurramendi, Y., Murga, R., et Kuijpers, C. (1996). Structure learning of bayesian networks by genetic algorithms : A performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9) :912–926.

[Lauritzen et Speigelhalter, 1988] Lauritzen, S. et Speigelhalter, D. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Royal statistical Society series B (Methodological)*, 50(2) :157–224.

[Lauritzen, 1995] Lauritzen, S. L. (1995). The EM algorithm for graphical association models with missing data. *Computational Statistics & Data Analysis*, 19(2) :191–201.

[Lauritzen, 1998] Lauritzen, S. L. (1998). *Graphical Models*, volume 17 of *Oxford Statistical Science Series*. Oxford Science Publications.

[Lauritzen et Wermuth, 1989] Lauritzen, S. L. et Wermuth, N. (1989). Graphical models for associations between variables, some of which are qualitative and some quantitative. *Annals of Statistics*, 17 :31Ű57.

[Leray, 2006] Leray, P. (2006). *Réseaux Bayésiens - Apprentissage et Modélisation de Systèmes Complexes*. Habilitation à diriger des recherches, Institut National des Sciences Appliquées de Rouen.

[Lerner et al., 2001] Lerner, U., Segal, E., et Koller, D. (2001). Exact inference in networks with discrete children of continuous parents. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 319–32, San Francisco, CA. Morgan Kaufmann.

[Li et D'Ambrosio, 1994] Li, Z. et D'Ambrosio, B. (1994). Efficient inference in bayes nets as a combinatorial optimization problem. *International Journal of Approximate Reasoning*, 11(1) :55–81.

[Lin, 1991] Lin, J. (1991). Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1) :145–151.

[Ling et Zhang, 2002] Ling, C. X. et Zhang, H. (2002). The representational power of discrete bayesian networks. *Journal of Machine Learning Research*, 3 :709–721.

[Lozano et al., 2006] Lozano, J. A., Larranaga, P., et Inza, I. (2006). *Towards a New Evolutionary Computation : Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing)*. Springer.

[MacKay, 1998] MacKay, D. J. C. (1998). Introduction to Monte Carlo methods. In Jordan, M. I., editor, *Learning in Graphical Models*, NATO Science Series, pages 175–204. Kluwer.

[Madigan et York, 1995] Madigan, D. et York, J. (1995). Bayesian graphical models for discrete data. *Int. Stat. Rev.*, 63 :215–232.

[Mahfoud, 1992] Mahfoud, S. W. (1992). Crowding and preselection revisited. In *Parallel Problem Solving from Nature 2, PPSN-II, Brussels, Belgium*, pages 27–36.

[Mahfoud, 1994] Mahfoud, S. W. (1994). Crossover interactions among niches. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, volume 1, pages 188–193, Piscataway, NJ. IEEE Service Center.

[Mahfoud, 1995] Mahfoud, S. W. (1995). *Niching methods for genetic algorithms*. PhD thesis, University of Illinois at Urbana-Champaign, Urbana, IL, USA. IlliGAL Report 95001.

[Margaritis, 2005] Margaritis, D. (2005). Distribution-free learning of bayesian network structure in continuous domains. In *AAAI*, pages 825–830.

[Martin et al., 1997] Martin, W. N., Lienig, J., et Cohoon, J. P. (1997). Island (migration) models : Evolutionary algorithms based on punctuated equilibria. In *Handbook of Evolutionary Computation.*, pages C6.3 :1–C6.3 :16. Oxford University Press.

[Masek, 2003] Masek, L. (2003). Recognition of human iris patterns for biometric identification.

[Masek et Kovesi., 2003] Masek, L. et Kovesi., P. (2003). Matlab source code for a biometric identification system based on iris patterns. Technical report, The School of Computer Science and Software Engineering, The University of Western Australia.

[McCallum et Nigam, 1998] McCallum, A. et Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *Proceedings of the AAAI/ICML-98 Workshop on Learning for Text Categorization*, pages 41–48. AAAI Press.

[Meek, 1997] Meek, C. (1997). *Graphical Models : Selecting causal and statistical models*. PhD thesis, Carnegie Mellon University.

[Meganck et al., 2006a] Meganck, S., Leray, P., Maes, S., et Manderick, B. (2006a). Apprentissage des réseaux bayésiens causaux à partir de données d'observation et d'expérimentation. In *Proceedings of 15ème Congrès Francophone Reconnaissance des Formes et Intelligence Artificielle, RFIA 2006*, page 131, Tours, France.

[Meganck et al., 2006b] Meganck, S., Leray, P., et Manderick, B. (2006b). Learning causal bayesian networks from observations and experiments : A decision theoritic approach. In *Proceedings of the Third International Conference, MDAI 2006*, volume 3885 of *Lecture Notes in Artificial Intelligence*, pages 58–69, Tarragona, Spain. Springer.

[Meganck et al., 2007] Meganck, S., Leray, P., et Manderick, B. (2007). Causal graphical models with latent variables : Learning and inference. In *Ninth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty ECSQARU 2007*, pages 5–16.

[Metropolis et al., 1953] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., et Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21 :1087–1092.

[Monro et al., 2007] Monro, D. M., Rakshit, S., et Zhang, D. (2007). DCT-based iris recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4) :586–595.

[Mühlenbein, 1991] Mühlenbein, H. (1991). Evolution in time and space - the parallel genetic algorithm. In Rawlins, G. J., editor, *Foundations of genetic algorithms*, page 316Ű337. Morgan Kaufmann, San Mateo, CA.

[Mühlenbein, 1998] Mühlenbein, H. (1998). The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5(3) :303–346.

[Mühlenbein et PaaB, 1996] Mühlenbein, H. et PaaB, G. (1996). From recombination of genes to the estimation of distributions. *Lecture Notes in Computer Science : Parallel Solving from Nature IV*, 1411 :178–187.

[Murphy, 2001] Murphy, K. (2001). The bayes net toolbox for matlab. *Computing Science and Statistics*, 33 :331–350.

[Murphy, 2003] Murphy, K. (2003). Active learning of causal bayes net structure. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–435.

[Muruzábal et Cotta, 2004] Muruzábal, J. et Cotta, C. (2004). A primer on the evolution of equivalence classes of bayesian-network structures. In *PPSN VIII : Parallel Problem Solving from Nature, 8th International Conference, Birmingham, UK, September 18-22,*, pages 612–621.

[Muruzábal et Cotta, 2007] Muruzábal, J. et Cotta, C. (2007). A study on the evolution of bayesian network graph structures. *Studies in Fuzziness and Soft Computing*, 213 :193–214,.

[Myers et al., 1999] Myers, J. W., Laskey, K. B., et De Jong, K. A. (1999). Learning bayesian networks from incomplete data using evolutionary algorithms. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., et Smith, R. E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO1999)*, volume 1, pages 458–465, Orlando, Florida, USA. Morgan Kaufmann.

[Naïm et al., 2004] Naïm, P., Wuillemin, P.-H., Leray, P., Pourret, O., et Becker, A. (2004). *Réseaux bayésiens*. Eyrolles, Paris.

[Nielsen et al., 2003] Nielsen, J. D., Kocka, T., et Peña, J. M. (2003). On local optima in learning bayesian networks. In *UAI '03, Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence, August 7-10 2003, Acapulco, Mexico*, pages 435–442.

[Parker, 2002] Parker, J. R. (2002). Genetic algorithms for continuous problems. In *Advances in Artificial Intelligence, 15th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2002, Calgary, Canada*, pages 176–184.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference.* Morgan Kaufmann, 1st edition.

[Pearl, 1997] Pearl, J. (1997). *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference.* Morgan Kaufmann, 2nd edition.

[Pearl, 2000] Pearl, J. (2000). *Causality : Models, Reasoning and Inference.* Cambridge. University Press, Cambridge, UK.

[Pearl et Verma, 1991] Pearl, J. et Verma, T. S. (1991). A theory of inferred causation. In Allen, J. F., Fikes, R., et Sandewall, E., editors, *KR'91 : Principles of Knowledge Representation and Reasoning*, pages 441–452, San Mateo, California. Morgan Kaufmann.

[Pelikan et al., 1999] Pelikan, M., Goldberg, D., et Cantú-Paz, E. (1999). BOA :the Bayesian Optimization Algorithm. In Banzhaf, W., Daida, J., Eiben, A., Garzon, M., Hovanar, J., Jakiela, M., et Smith, R., editors, *Proceedings of the genetic and evolutionary computation conference GECCO-99*, volume 1, pages 525–532, San Francisco, CA. Morgan Kaufmann.

[Perlman et Gillispie, 2001] Perlman et Gillispie (2001). Enumerating markov equivalence classes of acyclic digraphs models. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 171–177.

[Proenca et Alexandre, 2007] Proenca, H. et Alexandre, L. (2007). Toward noncooperative iris recognition : A classification approach using multiple signatures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4) :607–612.

[Proença et Alexandre, 2005] Proença, H. et Alexandre, L. A. (2005). Ubiris : A noisy iris image database. In *Proceedings of Image Analysis and Processing - ICIAP 2005, 13th International Conference, Cagliari, Italy, September 6-8, 2005*, pages 970–977.

[Radcliffe, 1991] Radcliffe, N. J. (1991). Equivalence class analysis of genetic algorithms. *Complex Systems*, 5(2) :183–205.

[Radcliffe, 1992] Radcliffe, N. J. (1992). Non-linear genetic representations. In *Parallel Problem Solving from Nature 2, PPSN-II, Brussels*, pages 261–270.

[Rechenberg, 1970] Rechenberg, I. (1970). *Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* PhD thesis, Technical University of Berlin, Berlin, Germany. (En allemand).

[Richter et Paxton, 2005] Richter, J. N. et Paxton, J. (2005). Adaptive evolutionary algorithms on unitation, royal road and longpath functions. In *Computational Intelligence*, pages 381–386.

[Rish, 2001] Rish, I. (2001). An empirical study of the naive bayes classifier. In *Proceedings of the IJCAI-01 Workshop on Empirical Methods in Artificial Intelligence.*

[Rissanen, 1978] Rissanen, J. (1978). Modelling by shortest data description. *Automatica*, 14 :465–471.

[Robert et Casella, 2004] Robert, C. P. et Casella, G. (2004). *Monte Carlo statistical methods*. Springer Texts in Statistics. Springer-Verlag, New York, second edition.

[Robinson, 1976] Robinson, R. (1976). Counting unlabeled acyclic digraphs. In *Combinatorial Mathematics V : Proceedings of the Fifth Australian Conference, held at the Royal Melbourne Institute of Technology, 1976*, pages 28–43. American Mathematical Society.

[Romero et al., 2004] Romero, T., Larrañaga, P., et Sierra, B. (2004). Learning bayesian networks in the space of orderings with estimation of distribution algorithms. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 18(4) :607–625.

[Rudolph, 1994] Rudolph, G. (1994). Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5(1) :96–101.

[Sahami et al., 1998] Sahami, M., Dumais, S., Heckerman, D., et Horvitz, E. (1998). A bayesian approach to filtering junk e-mail. In *Papers from the AAAI-98 Workshop on Text Categorization*, pages 55–62, Madison, WI.

[Saravanan et al., 1995] Saravanan, N., Fogel, D. B., et Nelson, K. M. (1995). A comparison of methods for self-adaptation in evolutionary algorithms. *BioSystems*, 36 :157–166.

[Schwartz, 1978] Schwartz, G. (1978). Estimating the dimensions of a model. *The Annals of Statistics*, 6(2) :461–464.

[Sebag et Ducoulombier, 1998] Sebag, M. et Ducoulombier, A. (1998). Extending population-based incremental learning to continuous search spaces. In *Parallel Problem Solving from Nature- PPSN V*, page 418Ű427, Berlin. Springer-Verlag.

[Sebag et al., 1998] Sebag, M., Schoenauer, M., et Peyral, M. (1998). Revisiting the memory of evolution. *Fundamenta Informaticae*, 35 :125–162.

[Spiessens et Manderick, 1991] Spiessens, P. et Manderick, B. (1991). A massively parallel genetic algorithm : Implementation and first analysis. In Belew, R. et Booker, L., editors, *Proceedings of the Fourth International Conference on Genetic Algorithms, San Mateo, CA*, page 279Ű286. Morgan Kaufman.

[Spirtes et al., 1993] Spirtes, Glymour, et Scheines (1993). *Causation, Prediction and Search.* Springer-Verlag.

[Spirtes et al., 2000] Spirtes, Glymour, et Scheines (2000). *Causation, Prediction and Search. (2d ed.)*. The MIT Press, 2nd edition.

[Spirtes et Scheines, 1991] Spirtes, P. Glymour, C. et Scheines, R. (1991). An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9(9) :62–72.

[Spirtes et al., 1999] Spirtes, P., Meek, C., et Richardson, T. (1999). An algorithm for causal inference in the presence of latent variables and selection bias. In *Computation, Causation, and Discovery*, pages 211–252. AAAI Press, Menlo Park, CA.

[Spirtes, 2001] Spirtes, R. (2001). An anytime algorithm for causal inference. In *Proceedings of the Conference on Artificial Intelligence and Statistics*.

[Sun, 2006] Sun, Z. (2006). Casia-irisv3.

[Surry et Radcliffe, 1997] Surry, P. D. et Radcliffe, N. J. (1997). Real representations. In Belew, R. K. et Vose, M. D., editors, *Foundations of Genetic Algorithms 4*, pages 343–363. Morgan Kaufmann, San Francisco, CA.

[Suzuki, 1996] Suzuki, J. (1996). Learning bayesian belief networks based on the minimum description length principle : An efficient algorithm using the b & b technique. In *International Conference on Machine Learning*, pages 462–470.

[Tanese, 1989] Tanese, R. (1989). Distributed genetic algorithms. In *Proceedings of the 3rd International Conference on Genetic Algorithms, George Mason University, Fairfax, Virginia, USA, June 1989*, pages 434–439.

[Thierens, 2002] Thierens, D. (2002). Adaptive mutation rate control schemes in genetic algorithms. Technical Report UU-CS-2002-056, Institute of Information and Computing Sciences, Utrecht University.

[Tsamardinos et al., 2006] Tsamardinos, I., Brown, L. E., et Aliferis, C. F. (2006). The max-min hill-climbing bayesian network structure learning algorithm. *Machine Learning*, 65(1) :31–78.

[Turing, 1948] Turing, A. (1948). Intelligent machinery. In Meltzer, B. et Michie, D., editors, *Machine Intelligence*, volume 5. Edinburgh University Press, Edinburgh.

[van Dijk et Thierens, 2004] van Dijk, S. et Thierens, D. (2004). On the use of a non-redundant encoding for learning bayesian networks from data with a ga. In *PPSN VIII : Proceedings of the 8th International Conference on Parallel Problem Solving from Nature, Birmingham, UK, September 18-22*, pages 141–150.

[van Dijk et al., 2003a] van Dijk, S., Thierens, D., et van der Gaag, L. (2003a). Building a ga from design principles for learning bayesian networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2003)*, pages 886–897.

[van Dijk et al., 2003b] van Dijk, S., van der Gaag, L. C., et Thierens, D. (2003b). A skeleton-based approach to learning bayesian networks from data. In *7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003*, pages 132–143.

[Vekaria et Clack, 1998] Vekaria, K. et Clack, C. (1998). Selective crossover in genetic algorithms : An empirical study. In *PPSN V : Proceedings of the 5th International Conference on Parallel Problem Solving from Nature, Amsterdam, The Netherlands, September 27-30, 1998*, pages 438–447.

[Verma et Pearl, 1990] Verma, T. et Pearl, J. (1990). Equivalence and synthesis of causal models. In *Proceedings of the Sixth Conference on Uncertainty and Artificial Intelligence*, pages 220–227. M. Kaufmann.

[Whitley, 1991] Whitley, L. D. (1991). Fundamental principles of deception in genetic search. In *Proceedings of the First Workshop on Foundations of Genetic Algorithms. Bloomington Campus, Indiana, USA, July 15-18 1990.*, pages 221–241.

[Whitley, 1994] Whitley, L. D. (1994). A genetic algorithm tutorial. *Statistics and Computing*, 4 :65–85.

[Wildes, 1997] Wildes, R. P. (1997). Iris recognition : an emerging biometric technology. *Proceedings of the IEEE*, 85(9) :1248–1363.

[Wolpert et Macready, 1995] Wolpert, D. H. et Macready, W. G. (1995). No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM.

[Wong et al., 1999] Wong, M., Lam, W., et Leung, K. S. (1999). Using evolutionary programming and minimum description length principle for data mining of bayesian networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(2) :174–178.

[Wong et al., 2002] Wong, M., Lee, S. Y., et Leung, K. S. (2002). A hybrid data mining approach to discover bayesian networks using evolutionary programming. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002)*, pages 214–222.

[Wright, 1921] Wright, S. (1921). Correlation and causation. *Journal of Agricultural Research*, 20 :557–585.

[Wright, 1964] Wright, S. (1964). Stochastic processes in evolution. In Gurland, J., editor, *Stochastic models in medecine and biology*, pages 199–241. University of Wisconsin Press, Madison, WI.

[Yu et al., 2002] Yu, J., Smith, V. A., Wang, P. P., Hartemink, A. J., et Jarvis., E. D. (2002). Using bayesian network inference algorithms to recover molecular genetic regulatory networks. In *International Conference on Systems Biology (ICSB02)*.

[Zaharie, 2004] Zaharie, D. (2004). A multipopulation differential evolution algorithm for multimodal optimization. In Matousek, R. et Osmera, P., editors, *Proceedings of Mendel, 10th International Conference on Soft Computing, Brno, june 2004*, pages 17–22.

[Zhang, 2006] Zhang, J. (2006). *Causal Inference and Reasoning in Causally Insufficient Systems.* PhD thesis, Carnegie Mellon University.

[Zhang et al., 2006] Zhang, J., Huang, D.-S., Lok, T.-M., et Lyu, M. R. (2006). A novel adaptive sequential niche technique for multimodal function optimization. *Neurocomputing*, 69(16-18) :2396–2401.

[Zhang, 2003] Zhang, N. (2003). Structural EM for hierarchical latent class models. Technical Report HKUST-CS03-06, Hong Kong University of Science & Technology.

[Zhigljavsky, 1991] Zhigljavsky, A. A. (1991). *Theory of global random search*. Kluwer Academic.

Annexe A

Probabilités et statistiques

A.1 Probabilités

Cette partie a pour objectif d'introduire les notions et propriétés de la théorie des probabilités employées dans nos travaux. Cette étude n'est pas exhaustive et nous limitons volontairement notre champ aux éléments nécessaires à la compréhension et à la manipulation des modèles étudiés. C'est pour cela que seules seront abordées les probabilités définies sur un espace discret et fini. Soit Ω, l'espace des observables (appelé aussi univers ou *espace des événements* dans la littérature), un ensemble fini non vide.

Définition 25 (tribu des événements)
Soit $\mathcal{A}$, un sous ensemble des parties de Ω, $\mathcal{A}$ a une structure de tribu s'il satisfait :
- *si $A \in \mathcal{A}$ alors son complémentaire $A^c = \Omega \backslash A$ est aussi dans $\mathcal{A}$.*
- *Soit une suite $A_1, A_2, \ldots, A_n$ finie et dénombrable d'éléments de $\mathcal{A}$, leur réunion $\bigcup_{n\geq 1} A_n$ est aussi dans $\mathcal{A}$.*
- *L'ensemble vide $\emptyset$ est dans $\mathcal{A}$.*

On appelle *événements* les éléments de $\mathcal{A}$.

Définition 26 (probabilité) *Soit un espace Ω d'observables et une tribu d'événements $\mathcal{A}$ formée de sous-ensembles de Ω, on appelle probabilité une fonction P de $\mathcal{A}$ dans $[0, 1]$ telle que :*
- *L'événement certain est de probabilité 1 : $P(\Omega) = 1$.*
- *Toute suite $A_1, A_2, \ldots, A_n$ d'événements de $\mathcal{A}$, deux à deux disjoints alors la série*

$$\sum_{i=1}^{\infty} P(A_i)$$

converge et a pour somme $P(\bigcup_{i\geq 1} A_i)$.

On appelle alors *espace de probabilités* ou *espace probabilisé* le triplet $(\Omega, \mathcal{A}, P)$.

Définition 27 (Variable aléatoire) *Soit $\{\Omega, \mathcal{A}, P\}$, un espace probabilisé et $\mathcal{B}$ la tribu des boréliens de $\mathbb{R}$.*
Une application :

$$X : \left\{ \begin{array}{ccc} \{\Omega, \mathcal{A}\} & \rightarrow & \{\mathbb{R}, \mathcal{B}(\mathbb{R})\} \\ \omega & \rightarrow & X(\omega) \end{array} \right.$$

est appelée variable aléatoire (ou v.a.) sur $\{\Omega, \mathcal{A}\}$ si :

$$\forall B \in \mathcal{B}(\mathbb{R}), (X \in B) \textit{ implique } X^{-1}(B) \in \mathcal{A}$$

Dans le cadre de notre étude, l'espace de définition des variables est dénombrable et fini ; par conséquent, nous travaillerons par la suite avec des variables aléatoires discrètes.

Définition 28 (loi d'une variable aléatoire discrète X)

$$p(x) = P(\omega \in \Omega | X(\omega) = x)$$

Propriété 1

- $0 \leq p(x) \leq 1 \quad \forall x \in \Omega'$
- $\sum_{x \in \Omega} p(x) = 1$
- $P(X(A)) = \sum_{x \in A} p(x)$

Définition 29 (probabilité jointe)
Soient X et Y deux variables aléatoires définies sur le même espace d'événements Ω. On définit la probabilité jointe P_{AB} de ces deux v.a. par :

$$P_{AB} : \left| \begin{array}{l} (\Omega, \mathcal{A}) \times (\Omega, \mathcal{B}) \rightarrow [0, 1] \\ (a, b) \mapsto P_{AB}(a, b) = P(\{\omega \in \Omega | \mathcal{A}(\omega) = a \wedge \mathcal{B}(\omega) = b\}) \end{array} \right.$$

Ce qui peut se généraliser à un ensemble $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ de n v.a. :

$$P_{\mathcal{X}} : \left| \begin{array}{l} \otimes(\Omega, \mathcal{A}_{X_i}) \rightarrow [0, 1] \\ x = (x_1, x_2, \ldots, x_n) \mapsto P_{\mathcal{X}}(x) = P(\{\omega \in \Omega | \bigwedge_{i \in 1, \ldots, n} X_i(\omega) = x_i\}) \end{array} \right.$$

A.1.1 Probabilités conditionnelles

Définition 30 (Probabilité conditionnelle)
Soit un espace probabilisé $\{\Omega, \mathcal{A}, P\}$, la probabilité conditionnelle $P(A|B)$ d'un événement A conditionnellement à un événement B tel que $P(B) > 0$ est définie par :

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Si $P(B) = 0$ alors $P(A|B)$ n'est pas définie.

Proposition 1 (Règle d'inversion de Bayes)
Soit un espace probabilisé $\{\Omega, \mathcal{A}, P\}$*, et* A *et* B *deux événements de* $\mathcal{A}$ *tels que* $P(A) > 0$ *et* $P(B) > 0$ *alors :*

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

Cette propriété est à la base du procédé d'inférence. En effet, si nous cherchons à mesurer notre incertitude quant à une hypothèse H en connaissant à partir d'une observation (ou ensemble de données) o, nous pouvons écrire :

$$P(H|o) = \frac{P(o|H)P(H)}{P(o)}$$

- P(H|o) est la probabilité *a posteriori* ;
- P(H) est la probabilité *a priori* ;
- P(o|H) est la vraisemblance de l'observation ;
- P(o) est une constante telle que : $\sum_h P(H = h|o) = 1$ soit $P(o) = \sum_h P(o|h)P(o)$.

Prenons l'exemple suivant : soit une maladie M provoquant l'apparition d'un symptôme A chez 85% des personnes atteintes. Nous savons qu'une personne a une chance sur un million d'être atteinte par M. Le symptôme A, quant à lui, est présent chez 5% des patients visitant le service médical. La question est : quelle est la probabilité qu'une personne atteinte d'une symptôme A souffre de la maladie en question ?
Soit M l'événement *être atteint de la maladie* M et A l'événement *présenter le symptôme* A.

$$P(M|A) = \frac{P(A|M)P(M)}{P(A)} = \frac{0,85 \times 10^{-6}}{0,05} = 1,7 \times 10^{-5}$$

A.1.2 Indépendances conditionnelles : définitions et mesures

Le calcul d'une probabilité jointe sur un ensemble conséquent de variables aléatoires, même binaires, requiert un nombre exponentiel de calculs. Il convient alors de simplifier la démarche et pour cela, exploiter les indépendances conditionnelles sous-jacentes au domaine.

Définition 31 (Indépendance conditionnelle)
Soit un univers Ω *et un ensemble* $\mathcal{X}$ *de variables aléatoires. Soient les sous ensembles* X, Y *et* $Z \subset \mathcal{X}$*.* X *est indépendant de* Y *conditionnellement à* Z *(ou* $(X \perp\!\!\!\perp Y|Z)$*) si et seulement si* X, Y *et* Z *vérifient :*

$$(X \perp\!\!\!\perp Y|Z) \Leftrightarrow \begin{cases} P(X|Y,Z) = P(X|Z) \\ \textit{et } P(Y|X,Z) = P(Y|Z) \end{cases}$$

Définition 32 (Indépendance marginale)
Soit un univers Ω *et un ensemble* $\mathcal{X}$ *de variables aléatoires. Soient les sous ensembles* X *et* $Y \subset \mathcal{X}$ *tels que* $(X \perp\!\!\!\perp Y|Z)$*) :*

$$(X \perp\!\!\!\perp Y|Z) \Leftrightarrow \begin{cases} \forall x \in (\Omega, \mathcal{A}_x) \textit{ avec } P(x) \geq 0, P(Y|X = x) = P(Y) \\ \textit{et } \forall y \in (\Omega, \mathcal{A}_y) \textit{ avec } P(y) \geq 0, P(X|Y = y) = P(X) \end{cases}$$

Cette définition signifie que la connaissance de la valeur prise par X, connaissant Z, n'apporte aucune information quant à celle prise par Y et inversement pour Y et X connaissant Z. Ceci nous permet de retrouver les résultats vus en 2.3.4. L'indépendance conditionnelle permet alors la simplification de l'écriture et du calcul de la loi jointe. Reprenons l'exemple du diagnostic vu précédemment et adjoignons à nos observations celle d'un symptôme B inputable lui aussi à la maladie M. On sait que, chez un patient atteint de la maladie M, le fait de souffrir du symptôme A n'influe en rien sur la survenue du symptôme B. Donc $P(B|M,A) = P(B|M)$ et ceci est aussi vrai dans le cas où le patient ne souffre pas de la maladie M : $P(B|\overline{M},A) = P(B|\overline{M})$. La probabilité jointe $P(M,A,B)$ peut alors s'écrire :

$$P(M,A,B) = P(B|A,M) \times P(A|M) \times P(M) = P(B|M) \times P(A|M) \times P(M)$$

Alors qu'avant la simplification l'écriture de la probabilité jointe $P(M,A,B)$ exigeait 2^3-1 entrées, la deuxième écriture prenant en compte l'indépendance conditionnelle $(B \perp\!\!\!\perp A|M)$ ne nécessite plus que $2+2+1$ entrées.

A.2 Formules et notions liés à l'indépendance conditionnelle

A.2.0.1 Test du χ^2

Le test du χ^2 a plusieurs emplois possibles :
- Test d'adéquation ;
- Test d'homogénéité ;
- Test d'indépendance.

Nous nous intéressons à son emploi afin de déterminer si deux variables aléatoires sont indépendantes.

Soient deux variables aléatoires discrètes X_A et X_B. Soient r_A et r_B, leurs cardinalités respectives. Soient N_{ab} le nombre d'occurrences observées dans la base d'exemples D, constituée de N cas, de $\{X_A = a \wedge X_B = b\}$. N_a et N_b le nombre d'occurrences observées de $\{X_A = a\}$ et $\{X_B = b\}$, respectivement. E_{ab} représente l'effectif théorique des occurrences de $\{X_A = a \wedge X_B = b\}$.

Deux hypothèses :

H_0 **:** les deux variables X_A et X_B sont indépendantes et $P(X_A \wedge X_B) = P(X_A) \times P(X_B)$.

H_1 **:** X_A et X_B ne sont pas indépendantes.

Le test quantifie la distance entre la fréquence observable des événements considérés et la fréquence hypothétique. Un seuil limite fait office de critère de validation de l'hypothèse H_0 (en l'occurrence *les variables sont indépendantes*).

$$\chi^2 = \sum_{a,b} \frac{(N_{ab} - E_{ab})^2}{E_{ab}} \tag{A.1}$$

La statistique suit asymptotiquement à N une loi du χ^2 à $(r_A - 1) \times (r_B - 1)$ degrés de liberté : l'hypothèse H_0 est vérifiée avec un seuil de confiance α si et seulement si $\chi^2 < \chi^2(df, 1-\alpha)$.

Dans le cadre de l'établissement non plus d'une indépendance entre deux variables mais d'une indépendance conditionnelle faisant intervenir une troisième variable aléatoire X_C, les hypothèses à vérifier deviennent :

H_0 **:** les deux variables X_A et X_B sont indépendantes conditionnellement à X_C.

H_1 **:** X_A et X_B ne sont pas indépendantes conditionnellement à X_C.

Et la formule A.1 se réécrit :

$$\chi^2 = \sum_{a,b,c} \frac{(N_{abc} - E_{abc})^2}{E_{abc}} \tag{A.2}$$

Avec N_{abc} le nombre d'occurrences observées de $\{X_A = a \wedge X_B = b \wedge X_C = c\}$. N_{ac} et N_{bc} le nombre d'occurrences observées de $\{X_A = a \wedge X_C = c\}$ et $\{X_B = b \wedge X_C = c\}$, respectivement. $E_{abc} = \frac{N_{ac}}{N} \times \frac{N_{bc}}{N} \times \frac{1}{N} = \frac{N_{ac} \times N_{bc}}{N}$.

A.2.1 Entropie

Pour une variable aléatoire X_A dotée d'une distribution de probabilité $P(X_A)$, l'entropie de X_A s'exprime par :

$$H(X_A) = -\sum_{x_a} P(X_A) \log(P(X_A)) = E(-\log(P(X_A)))$$

L'entropie *conditionnelle* d'une variable aléatoire X_A étant donné la valeur prise par une variable aléatoire X_B exprime la quantité d'information nécessaire pour inférer X_A à partir d'X_B.

Entropie Croisée Soit E_C, l'entropie croisée de X_A et X_B étant donné X_A,X_B et X_C trois variables aléatoires

$$E_C(X_A, X_B|X_C) = \sum_{x_c} (P(X_C) \sum_{x_a, x_b} P(X_A, X_B|X_C) \log\left(\frac{P(X_A, X_B|X_C)}{P(X_A|X_C) \times P(X_B|X_C)}\right)$$

L'entropie est bien entendu nulle quand il n'y a pas d'incertitude.

Ceci vérifie $E_C(X_A, X_B|X_C) = H(X_A|X_C) - H(X_A|X_B, X_C)$ et est aussi connu sous le nom d'Information Mutuelle : $I_m(X_A, X_B)$.

Démonstration 3 *Dans le cas de trois variables aléatoires, X_A, X_B et X_C, nous avons :*

$$H(X_A|X_C) = -\sum_{x_a} \sum_{x_b} \sum_{x_c} P(X_A, X_B, X_C) \log(P(X_A|X_C))$$

$$H(X_A|X_B, X_C) = -\sum_{x_a} \sum_{x_b} \sum_{x_c} P(X_A, X_B, X_C) \log(P(X_A|X_B, X_C))$$

donc l'information mutuelle $I(X_A, X_B|X_C)$ *vaut :*

$$I_m(X_A, X_B|X_C) = \sum_{x_a}\sum_{x_b}\sum_{x_c} P(X_A, X_B, X_C)\log(\frac{P(X_A|X_B, X_C)}{P(X_A|X_C)})$$

or,

$$P(X_A|X_B, X_C) = \frac{P(x_i, yj, z_k)}{P(yj, z_k)} = \frac{P(x_i, yj|z_k)P(z_k)}{P(yj|z_k)P(z_k)} = \frac{P(x_i, yj|z_k)}{P(yj|z_k)}$$

d'où

$$I_m(X, Y|Z) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} P(x_i, y_j, z_k)\log(\frac{P(X_A, X_B|X_C)}{P(X_A|X_C)P(X_B|X_C)})$$

$$= \sum_{x_c} P(X_C)\sum x_a \sum x_b P(X_A, X_B|X_C)\log(\frac{P(X_A, X_B|X_C)}{P(X_A|X_C)P(X_B|X_C)})$$

Pour deux variables aléatoires X_A et X_B, l'information mutuelle dépend à la fois de X_A et de X_B. Nous quantifions (conceptuellement) la quantité d'information sur X_B contenue dans X_A (et inversement, l'information étant symétrique, la quantité d'information sur X_A contenue dans X_B).
Cette mesure permet de définir une sorte de distance entre la probabilité jointe $P(X_A, X_B)$ et le produit de probabilités marginales $P(X_A) \times P(X_B)$.
Si X_A et X_B sont indépendantes, alors $P(X_A, X_B) = P(X_A) \times P(X_B)$ et l'information mutuelle est nulle.

Dans notre cas, nous nous intéressons à $I(X_A, X_B|X_C)$. Cette expression n'est nulle que si X_A et X_B sont conditionnellement indépendantes, connaissant X_C.

A.2.2 Rapport de vraisemblance

En lieu et place du test du χ^2, on peut souhaiter employer le test du rapport de vraisemblance aussi connu sous le nom de test du G^2.

$$G^2(X_A, X_B) \quad = \quad \sum_{a,b} N_{ab} \times \log\left(\frac{(N_{ab} \times N}{N_a \times N_b}\right). \tag{A.3}$$

Nous voyons ici que la valeur de G^2 est proportionnelle à celle de l'entropie croisée :

$$G^2(X_A, X_B) \quad = \quad 2 \times E_C(X_A, X_B). \tag{A.4}$$

G^2 suit alors une loi de distribution du χ^2 à $(r_a - 1)(r_b - 1)\prod_{c\in C} r_c$ degrés de libertés.

A.2.3 Test de Mann-Whitney

Il s'agit, à la base, d'un test statistique non-paramétré permettant de déterminer, avec un certain degré confiance, s'il existe une différence significative entre deux échantillons supposément indépendants (on dit aussi que ce test est un test *d'identité* permettant de déterminer si les deux échantillons sont issus d'une même distribution). Typiquement, on va chercher à déterminer si les valeurs d'un échantillon sont significativement plus petites que celle d'un deuxième échantillon.

Le test de Mann-Whitney repose sur la mesure d'un paramètre U, calculé en fonction des rangs des différentes valeurs des deux ensembles, les unes par rapport aux autres. La distribution de U sous l'hypothèse H_0 est connue (H_0 : les deux échantillons ne sont pas homogènes).

Il existe plusieurs manières de calculer U mais nous n'en détaillerons qu'une seule, simple. Nous nous limitons au cas d'échantillons de taille modeste, bien que des techniques d'approximation permettent de traiter le cas d'échantillons de très grande taille.

Le calcul de U s'effectue comme suit :

Soit E_c^1 le premier échantillon, de taille n_1 dont on cherche à déterminer si ses valeurs sont significativement plus petites que celles contenues dans le deuxième échantillon, E_c^2, de taille n_2 :

Commencer par regrouper les valeurs issues des deux échantillons dans un seul ensemble, E_c. Ordonner les valeurs de E_c (sans perdre leur origine) puis calculer R, la somme des rangs occupés par les valeurs issues de E_c^1 dans E_c.

U vaut alors

$$U = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R \tag{A.5}$$

La lecture de tables appropriées permet alors, à partir de la valeur de U de déterminer la validité ou non de l'hypothèse nulle H_0.

A.3 Mesures de divergence entre deux distributions de probabilités

Il peut être utile de pouvoir définir une notion de distance entre deux distributions de probabilités définies sur un même domaine. C'est ce que permettent les mesures introduites dans cette section : les divergences de Kullback-Leibler et celle de Jensen-Shannon.

A.3.1 Divergence de Kullback-Leibler

Soit X_i une variable aléatoire.
P et Q deux distributions de probabilités.

La divergence de Kullback-Leibler entre deux lois de distributions respectives P et Q est définie par :

$$KL(P||Q) = \sum_{x_i} P(X_i) \log\left(\frac{P(X_i)}{Q(X_i)}\right) \tag{A.6}$$

Les propriétés de cette divergence sont les suivantes :
- Non symétrique
- égale à zéro si et seulement si $P = Q$
- Si $\exists x_i$ tel que $P(x_i) > 0$ et $Q(x_i) = 0$, alors la divergence est infinie
- L'entropie de P vaut $\log(r) - D(P||U)$ avec m, le nombre de cas contenus dans $\mathcal{D}$ et U, la distribution uniforme.

la non-symétrie de la divergence de Kullback-Leibler ainsi que sa convergence vers $+\infty$ si il existe x_i tel que $p(x_i) \neq 0$ et $q(x_i) = 0$ rendent son emploi en tant que mesure de dissimilarité problématique. On préfère souvent employer une extension de la λ-divergence [El-Yaniv et al., 1997] :

$$D_\lambda(P||Q) = \lambda KL(P|||\lambda P + (1-\lambda)Q) + (1-\lambda)KL(Q|||\lambda P + (1-\lambda)Q) \tag{A.7}$$

Une extension particulière est définie pour $\lambda = \frac{1}{2}$: la divergence de Jensen-Shannon.

A.3.2 Divergence de Jensen-Shannon

Soit X_i une variable aléatoire.
P et Q deux distributions de probabilités.

La divergence de Jensen-Shannon entre deux lois de distributions respectives P et Q est définie par :

$$JS(P||Q) = \frac{1}{2}\left(\sum_{x_i} P(X_i) \log\left(\frac{2P(X_i)}{P(X_i)+Q(X_i)}\right) + \sum_{x_i} Q(X_i) \log\left(\frac{2Q(X_i)}{P(X_i)+Q(X_i)}\right)\right)$$

Les propriétés de cette divergence, comparativement à celles de la divergence de Kullback-Leibler sont les suivantes [Lin, 1991] :
- symétrique ;
- égale à zéro si et seulement si $P = Q$;
- bornée par 1 ;
- respecte l'inégalité triangulaire.

Annexe B

Analyse de texture

L'analyse texturale s'intéresse à la distribution spatiale des intensités dans l'image.

Nous nous intéressons particulièrement à l'étude de la distribution des statistiques à l'ordre deux, telle qu'elle fut proposée dans [Haralick et al., 1973].

B.1 Fondement

L'information de texture, selon les travaux d'Haralick, est contenue dans les relations spatiales entre les niveaux de gris. La représentation de ces différentes relations est établie à l'aide d'une matrice de cooccurrence. L'emploi de telles méthodes est rarement conseillé dans le cadre de la segmentation d'image et est plutôt recommandé pour l'analyse et la reconnaissance de textures.

Néanmoins, l'approche par matrice de cooccurrence demeure populaire principalement du fait que la représentation offerte par ces matrices est aisément appréhendable par l'utilisateur car elle reflète bien l'approche humaine d'identification des textures.

B.1.1 Matrices de cooccurrence

Les matrices de cooccurrence servent à représenter des relations spatiales dans un espace délimité suivant un angle et une distance donnés.

Soit I, une image de dimensions (N_x, N_y) et dotée de N_g niveaux de gris.

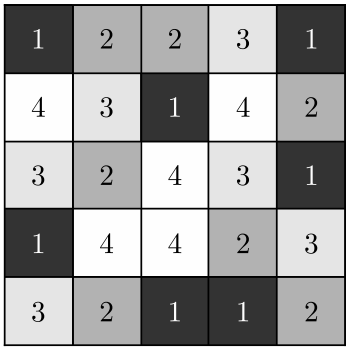
Une matrice de cooccurrence sur une telle image et pour une relation spatiale r définie par une direction ω et une distance d correspond à une matrice de dimension $N_g \times N_g$ dans laquelle chaque coefficient $M_{i,j}$, $i, j \in [0, \ldots, N_g - 1[^2$ se calcule comme suit :

$$M_{i,j} = \#\{(x,y),(x',y')\} \text{ tels que : } \begin{cases} (x' = x + d_x) \wedge (y' = y + d_y) \\ \text{et} \\ (I(x,y) = i \wedge I(x',y') = j) \vee (I(x,y) = j \wedge I(x',y') = i) \end{cases}$$

Avec :
– $I(x, y)$ le niveau de gris du pixel de I de coordonnées (x, y) ;
– $\#\{E\}$, notation désignant le nombre d'occurrences de l'événement E.

La matrice $M_{i,j}$ retranscrit dès lors le nombre de fois où l'on peut trouver, au sein de l'image, les transitions ($I(x, y) = i$ ou j) et ($I(x', y') = i$ ou j) suivant $r(\omega, d)$.

La figure B.1 montre un exemple de matrice d'occurrence à partir d'une image dotée de quatre niveaux de gris numérotés de 1 à 4 et de dimensions 5×5.



$$M_{r(0°,1)} = \begin{pmatrix} 2 & 3 & 3 & 2 \\ 3 & 2 & 3 & 3 \\ 3 & 3 & 0 & 2 \\ 2 & 3 & 2 & 2 \end{pmatrix}$$

Figure B.1 – Exemple de matrice de cooccurrence suivant une direction de 0° et une distance de 1 pixel.

B.1.2 Caractéristiques d'Haralick

Haralick *et al* ont défini 13 caractéristiques calculées à partir des matrices de cooccurrence. Les équations permettant le calcul de ces caractéristiques sont fournies dans le tableau B.2. Au préalable, il convient de calculer un ensemble de statistiques, présentées dans le tableau B.1. Soit :
– N_g, le nombre de niveaux de gris de l'image ;
– x et y, les coordonnées du point considéré ;
– $P(i, j)$, l'entrée de la matrice de cooccurrence correspondants aux niveaux de gris i et j.

$$R = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g}(ij)\,P(i,j) \tag{B.1}$$

$$p(i,j) = \frac{P(i,j)}{R} \tag{B.2}$$

$$p_x(i) = \sum_{j=1}^{N_g} p(i,j) \tag{B.3}$$

$$p_y(i) = \sum_{i=1}^{N_g} p(i,j) \tag{B.4}$$

$$p_{x+y}(k) = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j),\ (i+j=k), \quad k=2,3,\ldots 2N_g \tag{B.5}$$

$$p_{x-y}(k) = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j),\ |i-j=k|, \quad k=0,1,\ldots N_g-1 \tag{B.6}$$

$$HXY = -\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j)log\left(p(i,j)\right) \tag{B.7}$$

$$HXY1 = -\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j)log\left(p_x(i)p_y(j)\right) \tag{B.8}$$

$$HXY2 = -\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p_x(i)p_y(j)log\left(p_x(i)p_y(j)\right) \tag{B.9}$$

$$HX = -\sum_{i=1}^{N_g} p_x(i)log\left(p_x(i)\right) \tag{B.10}$$

$$HY = -\sum_{j=1}^{N_g} p_y(j)log\left(p_y(j)\right) \tag{B.11}$$

Tableau B.1 – Statistiques employées dans le calcul des caractéristiques d'Haralick

$$\text{Second Moment Angulaire} \quad f_1 = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j)^2 \tag{B.12}$$

$$\text{Contraste} \quad f_2 = \sum_{k=0}^{N_g-1} k^2 p_{x-y}(k) \tag{B.13}$$

$$\text{Corrélation} \quad f_3 = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_g}(ij)\,p(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y} \tag{B.14}$$

$$\text{Variance} \quad f_4 = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g}(i-\mu)^2 p(i,j) \tag{B.15}$$

$$\text{Moment de différence inverse} \quad f_5 = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g}\frac{p(i,j)}{1+(i-j)^2} \tag{B.16}$$

$$\text{Moyenne des sommes} \quad f_6 = \sum_{i=2}^{2N_g} i p_{x+y}(i) \tag{B.17}$$

$$\text{Variance des sommes} \quad f_7 = \sum_{i=2}^{2N_g}(i-f_8)^2\, p_{x+y}(i) \tag{B.18}$$

$$\text{Entropie des sommes} \quad f_8 = -\sum_{i=2}^{2N_g} p_{x+y}(i)\,log(p_{x+y}(i)) \tag{B.19}$$

$$\text{Entropie} \quad f_9 = -\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j)\,log(p(i,j)) \tag{B.20}$$

$$\text{Variance des différences} \quad f_{10} = Var(p_{x-y}) \tag{B.21}$$

$$\text{Entropie des différences} \quad f_{11} = -\sum_{i=0}^{N_g-1} p_{x-y}(i)\,log(p_{x-y}(i)) \tag{B.22}$$

$$\text{Info. sur les mesures de corrélation} \quad f_{12} = \frac{f_9 - HXY1}{max(HX,HY)} \tag{B.23}$$

$$\text{Coefficient de corrélation maximale} \quad f_{13} = \sqrt{1-exp^{-2(HXY2-f_9)}} \tag{B.24}$$

Tableau B.2 – Équations correspondant aux 13 caractéristiques d'Haralick

Annexe C

Résultats expérimentaux

Cette partie regroupe les résultats obtenus à l'issue de tests visant à déterminer les meilleurs paramétrages pour les différentes méthodes développées dans ce travail de thèse. L'ensemble des tests, notamment les réglages de paramètres relativement triviaux tels que les probabilités de croisement, de mutation ou encore la taille de la population pour l'algorithme panmictique ne sauraient être regroupés ici mais nous avons cependant pensé que le lecteur pourrait être intéressé par les résultats obtenus selon les valeurs prises par certains paramètres.

Les paramètres concernés sont :

Algorithme à stratégie de pénalisation : le nombre d'itérations Ite_{opt} effectuées avant mémorisation puis pénalisation d'une classe d'équivalence ;

Algorithme à stratégie de mémorisation : le coefficient γ par lequel sont multipliées les probabilités de mutation locales ;

Algorithme distribué : le nombre et la taille des populations considérées.

Les différents résultats ont été obtenus après 10 apprentissages sur des bases distinctes et mesurés auprès d'une unique base de 20000 cas.

C.1 Stratégie de pénalisation

Après quelques essais préliminaires, non reportés ici, nous avons décidé de tester plusieurs valeurs de Ite_{opt} et ce dans le cadre de l'apprentissage de la structure du réseau Insurance. Les résultats sont retranscrits dans le tableau, pour des valeurs de Ite_{opt} allant de 10 à 30. Au delà de cet intervalle, les performances de l'algorithme s'avèrent dégradées, y compris par rapport à la version simple, sans stratégie de pénalisation.

Les tests effectués ici l'ont été avec une population de 150 individus.

La qualité des solutions en termes de distance à la structure d'origine et de scores permet de dégager deux valeurs : 10 et 20. Sur la base de 250 cas, en particulier, une valeur de 10 pour

	INSURANCE			
	250	**500**	**1000**	**2000**
$Ite_{opt} = 10$	$-3189; 38,6$	**-3089 ; 33,1**	**-2937**$; 30,5$	$-2862; 25,1$
$Ite_{opt} = 15$	$-3194; 37,7$	$-3106; 33,3$	$-2951; 31,2$	$-2861; 27,2$
$Ite_{opt} = 20$	**-3176 ;37,0**	$-3106; 34,0$	$-2969; 30,8$	**-2856 ;24,4**
$Ite_{opt} = 25$	$-3193; 38,3$	-3108;**33,1**	-2947;**30,0**	$-2860; 27,1$
$Ite_{opt} = 30$	$-3190; 38,4$	$-3098; 33,8$	$-2956; 31,2$	$-2860; 24,6$

Tableau C.1 – Scores BIC moyens, divisés par 100 et arrondis, des structures obtenues pour le réseau Insurance et nombre d'arcs différents de la structure recherchée pour différentes valeurs du paramètre Ite_{opt}. Les scores affichés sont obtenus avec une base de 20000 cas d'exemples de test. Les meilleurs résultats apparaissent en gras.

le paramètre Ite_{opt} permet de recouvrir des réseaux très performants en matière de score mais, paradoxalement, les plus distants graphiquement du graphe d'origine.

Ceci signifierait, dans une moindre mesure, qu'une valeur de Ite_{opt} trop petite entraînerait l'apprentissage de réseaux trop spécialisés (situation de surapprentissage) en plus d'un surcoût de calcul à chaque itération due au parcours de la liste des optima connus. La valeur choisie pour la suite de nos expériences a dont été de $Ite_{opt} = 20$.

C.2 Stratégie d'adaptation de la mutation

Nous testons ici trois valeurs possibles pour le coefficient γ employé dans l'algorithme 7.2 :
- 0,25
- 0,50
- 0,75

Les tests effectués ici l'ont été, comme précédemment, sur l'apprentissage de la structure du réseau Insurance mais avec, ici, une population de 100 individus.

	INSURANCE			
	250	**500**	**1000**	**2000**
$\gamma = 0,25$	$-3207; 40,5$	-3098;**33,5**	**-2944 ;30,2**	$-2894; 30,2$
$\gamma = 0,50$	**-3204**$; 40,8$	**-3109**$; 34,5$	$-2966; 32,3$	-2873;**27,1**
$\gamma = 0,75$	-3208;**39,7**	**-3107**$; 33,9$	$-2950; 33,3$	$-2881; 29,7$

Tableau C.2 – Valeurs moyennes, divisées par 100 et arrondies des scores BIC des structures obtenues pour le réseau Insurance pour différentes valeurs du paramètre γ et nombre d'arcs différents de la structure recherché, pour des bases de tailles différentes (colonnes). Les scores affichés sont obtenus avec une base de 20000 cas d'exemples de test. Les meilleurs résultats apparaissent en gras.

Le tableau C.2 ne permet pas de distinguer clairement une valeur optimale pour le paramètre γ. Après observation des différentes matrices de coefficients employées par cette méthode, il s'avère que, du fait d'une probabilité de mutation élevée en conjugaison avec une population

assez nombreuse (150 individus), une valeur modeste de γ permet à elle seule une chute rapide des coefficients correspondant aux mutations dommageables. Inversement, le paramètre devant être assez élevé non seulement pour pouvoir promouvoir une mutation intéressante mais aussi pour pouvoir inverser une tendance (une mutation dommageable par le passé mais bénéfique à l'instant t en cours) nous avons opté pour une solution intermédiaire avec la valeur $\gamma = 0,5$.

C.3 Algorithme distribué

Lors de l'implémentation de la version distribuée de l'algorithme à stratégie de pénalisation, quatres paramètres supplémentaires restaient à déterminer (cf section 6.3) :

- l'intervalle migratoire ;
- le taux de migration ;
- le nombre de populations ;
- la taille des populations.

Nous avons testé l'apprentissage de la structure du réseau Insurance (cf chapitre 8) sur 10 bases d'apprentissage différentes avant de moyenner les résultats. Nous avons effectué les tests avec les valeurs suivantes pour nos paramètres :

intervalle migratoire : égal à 20 ou 40 itérations ;

taux de migration : égal à 10% ou 30% ;

nombre de populations : de 10 à 30 populations avec un incrément de 10 ;

taille des populations : de 10 à 40 individus avec un incrément de 10.

Les résultats suivants indiquent les scores obtenus, en moyenne, sur une unique base de test de 20000 cas ainsi que la distance graphique (nombre d'arcs différents) moyenne entre les réseaux obtenus et la structure d'origine.

Alors que les performances de l'algorithme sont croissantes avec le nombre et la taille des sous-populations, en toute logique, les valeurs de l'intervalle migratoire et du taux de migration ne paraissent pas avoir un rôle décisif dans les performances de l'heuristique. Afin de réduire les calculs, nous avons choisi d'établir les paramètres à 30 populations de 30 individus pour un intervalle migratoire de 20 itérations et un taux de migration de 10%. Le choix de valeurs basses pour ces deux derniers paramètres s'étant fait d'une part sur la volonté de pouvoir mettre en place un certains nombre de mouvements d'individus au court d'une instance et, d'autre part, sur le souhait de vouloir limiter l'impact d'une migration trop importante sur les populations d'accueil.

	INSURANCE			
	250	**500**	**1000**	**2000**
Nb(pop) ;taille(pop)				
10 ;10	−3217; 41, 6	**-3108**; 35, 7	−2978; 32, 7	−2904; 30, 0
10 ;20	−3226; 40, 7	−3128; 36, 2	−2972; 31, 3	−2883; 28, 1
10 ;30	−3219; 40, 6	−3114; 35, 6	−2972; 31, 6	−2892; 28, 8
10 ;40	−3218; 40, 0	−3116; 34, 7	−2971; 31, 1	−2885; 27, 4
20 ;10	−3213; 40, 6	−3122; 35, 7	−2993; 31, 0	−2901; 28, 2
20 ;20	**-3202**; 38, 8	−3114; 34, 4	−2955; 32, 2	−2865; 28, 2
20 ;30	−3203;**38,6**	−3124; 36, 0	−2942; 31, 4	−2857; 27, 4
20 ;40	−3206; 39, 4	−3110;**33,4**	−2948; 31, 0	−2859; 23, 9
30 ;10	−3215; 40, 3	−3123; 35, 8	−2968;**30,8**	−2872; 26, 6
30 ;20	−3209; 40, 1	**-3108 ;**35, 4	−2968; 32, 3	−2859; 25, 8
30 ;30	**-3202**; 39, 4	−3114; 34, 5	−2936; 31, 1	−2854; 25, 6
30 ;40	−3208; 39, 3	−3112; 34, 9	**-2935**; 32, 4	**-2848 ;22,9**

Tableau C.3 – Scores BIC obtenus par l'algorithme distribué pour différentes tailles de bases d'apprentissage pour le réseau Insurance. Les scores des réseaux obtenus ont été calculés et moyennés à partir d'une unique base de 20000 cas générée par échantillonnage du réseau d'origine. Les scores de ce tableau ont été obtenus pour une période migratoire de 40 itérations et un taux de migration de 10%. Les meilleurs résultats apparaissent en gras.

	INSURANCE			
	250	**500**	**1000**	**2000**
Nb(pop) ;taille(pop)				
10 ;10	−3228; 41, 1	−3138; 37, 4	−3014; 35, 0	−2921; 30, 6
10 ;20	−3220; 40, 0	−3114; 36.5	−2978; 33, 1	−2905; 30, 9
10 ;30	−3210; 39, 9	−3128; 36, 0	−2989; 32, 9	−2875; 30, 3
10 ;40	−3212; 40, 9	−3120; 35, 1	−2944; 32, 2	−2890; 29, 2
20 ;10	−3204; 39, 3	−3136; 36, 8	−2984; 32, 7	−2914; 31, 5
20 ;20	−3217; 40, 3	−3126; 35, 5	−2968; 31, 7	−2865;**25,2**
20 ;30	−3212; 40, 0	−3117; 35, 1	−2947; 31, 8	−2865; 27, 6
20 ;40	−3213; 40, 3	−3116; 34, 0	−2950; 31, 0	−2861; 26, 9
30 ;10	−3211; 40, 7	−3120; 36, 1	−2964; 31, 1	−2884; 28, 8
30 ;20	−3212; 40, 2	**-3106**; 34, 0	−2970;**30,8**	−2867; 25, 9
30 ;30	**-3201**; 38, 8	−3111;**33,9**	**-2933**; 31, 8	−2864; 26, 3
30 ;40	−3206; 40, 1	−3120; 36, 2	−2936; 32, 1	**-2859**; 28, 3

Tableau C.4 – Scores BIC obtenus par l'algorithme distribué pour différentes tailles de bases d'apprentissage pour le réseau Insurance. Les scores des réseaux obtenus ont été calculés et moyennés à partir d'une unique base de 20000 cas générée par échantillonnage du réseau d'origine. Les scores de ce tableau ont été obtenus pour une période migratoire de 40 itérations et un taux de migration de 30%. Les meilleurs résultats apparaissent en gras.

	INSURANCE			
	250	**500**	**1000**	**2000**
Nb(pop) ;taille(pop)				
10 ;10	−3217; 40, 4	−3134; 38, 0	−2967; 32, 6	−2895; 26, 9
10 ;20	−3208; 39, 3	−3131; 35, 2	−2976; 33, 5	−2884; 31, 0
10 ;30	−3203; 39, 5	−3115; 35, 1	−2970; 31, 5	−2869; 27, 3
10 ;40	−3194; 37, 7	−3117; 35, 6	−2942; 32, 4	−2863; 25, 8
20 ;10	−3203; 39, 8	−3120; 35, 2	−2977; 32, 5	−2884; 27, 8
20 ;20	−3208; 39, 3	−3112; 34, 8	−2955; 31, 0	−2868; 27, 5
20 ;30	−3194;**37,4**	−3097; 34, 5	−2932; 29, 5	**-2853**; 25, 0
20 ;40	−3207; 39, 1	−3109; 34, 2	−2934; 30, 1	−2855; 25, 6
30 ;10	−3202; 38, 6	−3118; 34, 8	−2938; 30, 2	−2860; 25, 4
30 ;20	−3197; 38, 3	−3106; 34, 0	**-2928**; 29, 7	−2857;**24,8**
30 ;30	**-3193**; 38, 1	**-3104 ;33,3**	−2934;**29,3**	−2860; 26, 6
30 ;40	−3197; 38, 8	−3107; 33, 7	**-2928**; 30, 9	−2857; 26, 2

Tableau C.5 – Scores BIC obtenus par l'algorithme distribué pour différentes tailles de bases d'apprentissage pour le réseau Insurance. Les scores des réseaux obtenus ont été calculés et moyennés à partir d'une unique base de 20000 cas générée par échantillonnage du réseau d'origine. Les scores de ce tableau ont été obtenus pour une période migratoire de 20 itérations et un taux de migration de 10%. Les meilleurs résultats apparaissent en gras.

	INSURANCE			
	250	**500**	**1000**	**2000**
Nb(pop) ;taille(pop)				
10 ;10	−3226; 42, 0	−3136; 37, 8	−3003; 34, 4	−2906; 30, 6
10 ;20	−3217; 39, 3	−3125; 37, 2	−2966; 32, 5	−2883; 28, 8
10 ;30	−3208; 39, 8	−3125; 35, 2	−2966; 32, 2	−2870; 27, 7
10 ;40	−3201; 37, 8	−3111; 34, 4	−2956; 29, 9	−2860; 26, 5
20 ;10	−3216; 39, 5	−3120; 37, 0	−2999; 32, 6	−2886; 28, 0
20 ;20	−3208; 39, 6	−3113; 34, 8	−2965; 30, 9	−2861; 26, 3
20 ;30	**-3191**; 37, 8	−3110; 35, 6	−2959; 30, 9	−2875; 25, 9
20 ;40	−3213; 39, 8	−3117; 34, 7	−2946; 32, 4	−2858; 24, 8
30 ;10	−3205; 38, 7	−3117; 35, 0	−2948; 31, 8	−2880; 30, 3
30 ;20	−3207; 38, 6	−3104; 34, 4	−2946; 30, 1	−2862; 27, 8
30 ;30	−3210; 39, 1	**-3097 ;33,2**	**-2942 ;29,7**	**-2854 ;23,6**
30 ;40	−3208; 40, 0	−3101; 33, 4	−2936; 30, 9	−2858; 24, 2

Tableau C.6 – Scores BIC obtenus par l'algorithme distribué pour différentes tailles de bases d'apprentissage pour le réseau Insurance. Les scores des réseaux obtenus ont été calculés et moyennés à partir d'une unique base de 20000 cas générée par échantillonnage du réseau d'origine. Les scores de ce tableau ont été obtenus pour une période migratoire de 20 itérations et un taux de migration de 30%. Les meilleurs résultats apparaissent en gras.

Table des figures

Liste des tableaux

Résumé :

Dans ce travail de thèse, nous proposons d'étudier le problème de l'apprentissage de la structure d'un réseau bayésien par un ensemble de méthodes évolutionnaires. Après avoir conçu un algorithme génétique parcourant l'espace des structures, nous avons élaboré différentes techniques visant à améliorer les performances de cet algorithme. Nous avons ainsi développé une stratégie de parcours visant à exploiter les propriétés de l'espace des graphes essentiels à travers un mécanisme de *niching* séquentiel, mécanisme que nous étendons par une hybridation avec une modélisation en îlots. Une autre méthode définit une distribution de probabilités sur les opérations de mutation appliquées à la population, déterminée par la qualité des individus modifiés.

Mots clés :

réseaux bayésiens, algorithme génétique, apprentissage de structure, détection de l'iris, caractéristiques de texture d'Haralick.

Abstract :

In this thesis, we propose a study of the problem of learning the structure of a bayesian network through the use of evolutionary methods. We first designed a genetic algorithm to search the space of structures before establishing various strategies aiming at improving the performances of this algorithm. We consequently developed a search strategy aiming at exploiting the properties of the space of completed partially oriented graphs using a sequential niching principle which we later hybridized with an island model scheme. Another method defines a distribution probability over the mutation operations which are applied to the individuals and that is a function of the qualitative results of previously applied operations.

Keywords :

bayesian networks, genetic algorithms, structure learning, iris detection, Haralick texture features.

Université François-Rabelais de Tours, Laboratoire d'Informatique, EA 2101, Équipe Reconnaissance des Formes et Analyse de l'Image (`http://www.li.univ-tours.fr`). Polytech'Tours, Département Informatique, 64 Avenue Jean Portalis, 37200 Tours (`http://www.polytech.univ-tours.fr`).