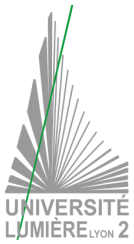
Université Lumière - Lyon II

Rapport de stage de Master Recherche en Informatique
Spécialité Extraction de Connaissances à partir des Données
Année 2005-2006

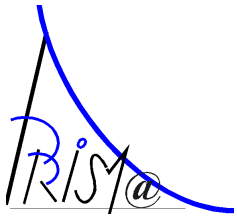
Algorithmes génétiques pour l'identification structurelle des réseaux bayésiens

Stagiaire :
Jean-Marc Trémeaux

Encadrant :
Jean-Marc Adamo (PRISMa)



Université Lumière - Lyon II



PRoductique et Informatique
des Systèmes Manufacturiers

Résumé

Les réseaux bayésiens constituent un corps de techniques puissantes pour la mise en forme et la mise en œuvre de modèles d'aide à la décision. L'identification structurelle de ces réseaux à partir de données est un problème de forte complexité algorithmique, encore mal résolu à ce jour pour des jeux de données de grande taille. Les algorithmes génétiques apportent de bonnes solutions approchées dans ce domaine. Après avoir analysé et classé les approches décrites dans la littérature, je propose deux algorithmes d'apprentissage structurel de réseaux bayésiens, puis les réalise et étudie leur comportement sur différents jeux de données.

Mots-Clés

Réseaux bayésiens, Algorithmes génétiques, Apprentissage automatique, Optimisation combinatoire

Table des matières

1 Introduction

Les réseaux bayésiens sont un outil de modélisation à la jonction de la théorie des probabilités et de la théorie des graphes, permettant de décrire les relations régissant un ensemble variables aléatoires et d'effectuer un raisonnement probabiliste sur celles-ci. Il constitue à la fois un formalisme de représentation des connaissances, ainsi qu'un outil permettant d'expliquer et de prédire l'état de variables d'intérêt d'un domaine de connaissances en fonction de l'état de variables observées. De part le développement d'algorithmes d'inférences efficaces, les réseaux bayésiens ont été récemment mis en applications dans de nombres domaines, tels que l'aide au diagnostic médical et industriel, la surveillance de réseaux de télécommunications, la classification automatique de documents structurés ou encore l'analyse d'images.

Une difficulté majeure de la mise en forme de réseaux bayésiens est celle de l'élicitation de modèles graphiques et de leurs paramètres associés par un expert du domaine concerné. La spécification de modèles de taille réaliste est en effet souvent trop complexe pour être réalisée par un agent humain. L'approche moderne, née du faible coût du stockage et de l'exploitation de vastes bases de données, est l'apprentissage automatique de réseaux bayésiens à partir de données préalablement collectées.

L'apprentissage se décompose naturellement en deux phases indépendantes. La première phase consiste à identifier la structure du modèle, c'est-à-dire un graphe orienté acyclique. La seconde phase est l'estimation des paramètres, c'est-à-dire des tables de probabilités conditionnelles associées à chaque sommet.

Je m'intéresse ici au problème d'identification du modèle graphique représentant au mieux la structure des données d'apprentissage. Ce problème d'apprentissage non supervisé est de forte complexité algorithmique, ce qui implique qu'il est impossible dans le cas général de trouver sa solution exacte dans un temps raisonnable. Une solution possible est de se restreindre à des modèles simples, tels que des arbres, ou des *réseaux bayésiens naïfs*, mis en œuvre dans certains problèmes de prédiction. Cependant, un modèle contraint de telle sorte porte le risque de mal refléter la structure intrinsèque aux données étudiées. Une autre approche au problème de la complexité algorithmique est, à défaut de méthodes plus efficaces, de développer des algorithmes d'optimisation combinatoire permettant, dans un espace très vaste de modèles candidats, d'identifier une solution s'approchant au mieux du modèle optimal.

Ces algorithmes approchés, dits « méta-heuristiques », sont des méthodes de recherche stochastique dans un espace d'hypothèses procédant par échantillonage d'une fonction objectif jusqu'à converger vers un optimum. Ils sont le plus souvent inspirés de processus biologiques ou physiques. Parmi les méthodes les plus populaires, on peut citer les *al-*

gorithmes génétiques ou la *programmation évolutionnaire*, les algorithmes de *colonies de fourmis*, le *recuit simulé* ou encore les *systèmes immunitaires artificiels*. La classe des algorithmes génétiques constitue une piste de recherche intéressante et bénéficiant d'une littérature abondante pour l'identification structurelle de réseaux bayésiens.

Les algorithmes génétiques sont inspirés des phénomènes de l'évolution naturelle. Ils procèdent par l'évaluation d'une population d'« individus » codant les modèles candidats, par analogie au « matériel génétique ». Par des processus itératifs de sélections, recombinaisons et mutations, ces individus sont amenés à s'adapter progressivement au problème à résoudre. Les individus conservés au cours du temps sont ceux qui comportent les « gènes » adaptés, afin de satisfaire au mieux la fonction objectif. Un intérêt majeur est qu'il n'est pas nécessaire d'introduire de connaissances particulières (ou heuristiques) sur la résolution du problème, mais seulement de pouvoir formuler et échantilloner une fonction objectif. L'approche est donc très générale et appropriée à la résolution de tout problème dit « complexe », mais peuvent cependant être spécialisée si besoin est par ajout d'opérateurs de recherche locale. Les problématiques majeures de l'mplémentation d'algorithmes génétiques sont de pouvoir exprimer un codage adapté des hypothèses, des opérateurs de recombinaison et de mutation, ainsi que de choisir les conditions initiales et les paramètres d'apprentissage : taille de la population, fréquence d'application des opérateurs, *etc.* Ces spécificités d'application des paradigmes de l'algorithmique génétique au problème particulier de l'identification structurelle de réseaux bayésiens sont l'objet de ce travail.

Après avoir énoncés les principes des réseaux bayésiens et de leur mise en forme, je présenterai les algorithmes génétiques et j'étudierai leur application à l'apprentissage automatique de structure des réseaux bayésiens. Puis, je décrirai la mise en œuvre de deux de ces algorithmes dans le logiciel libre de fouille de données *Weka*. Enfin, j'évaluerai les performances de ces algorithmes pour l'apprentissage de deux modèles, l'un artificiel et l'autre réel. Je présenterai divers résultats avant de conclure.

2 Réseaux bayésiens

2.1 Généralités

Les réseaux bayésiens sont utilisés pour décrire et raisonner sur les situations où divers attributs sont liés par des chaînes d'inférence. La figure 1 et la table 1, tirées de l'ouvrage de Neapolitan [Nea03], constituent un exemple d'un tel réseau. Dans celui-ci, les antécédents de fumeur (H) ont une *influence directe* sur la possibilité de développer une bronchite (B) ou un cancer des poumons (L). La présence ou l'absence de telles maladies ont à leur tour une influence directe sur la possibilité de ressentir de la fatigue (F) ou d'avoir un examen aux rayons X positif (C). L'absence d'arc entre H et F exprime le fait qu'un antécédent de fumeur a une influence sur la fatigue uniquement par le biais de la présence d'une maladie. En revanche, la bronchite n'a, de part la structure du graphe, aucune influence (directe ou indirecte) sur un l'examen aux rayons X. Ce modèle permet donc de structurer et de décrire un domaine de connaissances médical ainsi que de raisonner sous l'incertitude, à savoir calculer, après avoir observé l'état d'un ensemble de descripteurs, l'état probable d'attributs dont l'observation n'est pas possible.

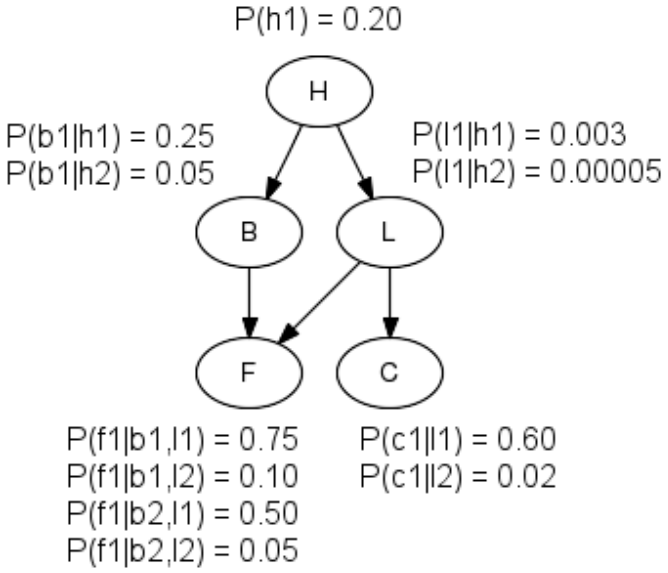


FIG. 1 – *Exemple de réseau bayésien (structure et tables de probabilités conditionnelles)*

Attribut	Modalité	Commentaire
H	$h1$	Antécédent fumeur
	$h2$	Antécédent non fumeur
B	$b1$	Bronchite présente
	$b2$	Bronchite absente
L	$l1$	Cancer des poumons présent
	$l2$	Cancer des poumons absent
F	$f1$	Fatigue présente
	$f2$	Fatigue absente
C	$c1$	Rayons X des poumons positifs
	$c2$	Rayons X des poumons négatifs

TAB. 1 – Modalités du réseau bayésien de la figure 1

2.2 Définitions

Un réseau bayésien est un graphe orienté acyclique, dans lequel chaque sommet correspond à une variable aléatoire du domaine. Un arc $X \rightarrow Y$, décrit une relation père-fils dans laquelle X est le père et Y le fils. L'ensemble des parents du sommet X est noté Π_X. De plus, à chaque sommet est associé une table de probabilités conditionnelles, spécifiant la probabilité de chaque état du sommet étant donné la combinaison d'états de ses parents.

Définition 1 *Soient X, Y et Z trois ensembles disjoints de variables aléatoires. Alors X est indépendant de Y conditionnellement à Z, si et seulement si $p(x|z,y) = p(x|z)$ pour toutes les valeurs possibles x, y et z des variables X, Y et Z. On note alors $I(X,Y|Z)$. Une condition d'indépendance est caractérisée par son ordre, c'est-à-dire le nombre de variables contenues dans l'ensemble Z.*

Définition 2 : Condition de Markov *Soit un ensemble V de variables aléatoires, une distribution jointe P sur V et un graphe orienté acyclique $G = (V,E)$. Alors le couple (G,P) satisfait la condition de Markov si et seulement si, pour chaque variable $X \in V$, X est indépendante de l'ensemble de ses non-descendants dans le graphe conditionellement à l'ensemble de ses parents, ie. $I_P(X, ND_X|\Pi_X)$.*

Définition 3 : Réseau bayésien *Soient V un ensemble de variables aléatoires et P une distribution jointe sur V.*

Soit $G = (V,E)$ un graphe orienté acyclique.

Alors (G, P) est un réseau bayésien si et seulement si (G, P) satisfait la condition de Markov.

Par conséquence, un réseau bayésien code une distribution jointe de probabilité des variables du domaine de la façon suivante :

$$P(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | \Pi_{X_i})$$

Cette propriété permet effectivement de coder et de calculer la distribution jointe sur V avec un nombre réduit de paramètres en supposant que les conditions d'indépendance soient vérifiées. En supposant que la taille des tables de probabilités conditionnelles est constante, la complexité du modèle est alors linéaire en fonction du nombre de sommets alors qu'elle est exponentielle dans la distribution jointe.

2.3 Apprentissage automatique de réseaux bayésiens

Comme il a été montré dans le paragraphe précédent, une fois un réseau bayésien spécifié, celui-ci constitue un outil efficace pour effectuer une inférence probabiliste. Néanmoins, le problème de la construction des modèles demeure. Ceux-cis peuvent être spécifiés par un des experts du domaine, mais cette approche est sujette aux erreurs et ne permet pas de construire des modèles de taille conséquente. Une autre approche est d'inférer un modèle à l'aide d'un algorithme et d'une base d'exemples.

Les algorithmes d'apprentissage se divisent en deux catégories. La première fait usage de tests d'indépendance d'ordre croissant, et la seconde de recherche dans un espace d'hypothèses à l'aide d'une mesure de qualité [vDvdGT03]. Parmi les méthodes de recherche, les recherches dites *gloutonnes* prennent une suite de décisions locales optimales, alors que la famille dite des *méta-heuristiques* font usage de recherche stochastique afin d'explorer un espace plus vaste.

2.3.1 Analyse des dépendances

Cette méthode vise à découvrir les relations causales dans les données. Elle suppose qu'il existe une structure de réseau qui représente exactement les conditions d'indépendances de la distribution de variables aléatoire qui a généré les données. L'analyse des dépendances procède par tests statistiques d'ordre croissant tels que la statistique du χ^2 ou le *critère d'information mutuelle*. Pour chaque relation d'indépendance conditionnelle identifiée, il s'ensuit que le graphe ne contient pas d'arc entre les sommets concernés.

Cette approche possède un inconvénient majeur : la quantité de données nécessaires à un test est une fonction exponentielle de son ordre. Les tests d'ordre élevé deviennent non-fiables, et cette technique est donc en général seulement utilisée pour les tests d'ordre 0 et 1.

2.3.2 Recherche gloutonne

Un principe couramment utilisé en fouille de données (apprentissage de *graphes d'induction*, *clustering*, *etc.*) est la recherche locale avec heuristique. L'équivalent direct pour l'apprentissage de structure de réseaux bayésiens est l'algorithme $K2$ de Cooper et Herskovits [CH92]. Celui-ci permet, étant donné un jeu de données D, de chercher la structure B_S du réseau bayésien qui maximise $P(B_S, D)$.

Algorithme 1 Algorithme K2

Entrées: Un ensemble de n sommets, un ordre total sur ces sommets, une borne supérieure u au nombre de parents qu'un sommet peut avoir, un ensemble D contenant m exemples.

Sorties: Pour chaque sommet, l'ensemble des parents de ce sommet.

pour $i = 1$ à n **faire**
  $\Pi_i \leftarrow \emptyset$
  $P_{old} \leftarrow g(i, \Pi_i)$
  $Continuer \leftarrow VRAI$
  tantque $Continuer$ ET $|\Pi_i| < u$ **faire**
    $Z \leftarrow$ sommet de $Pred(X_i) - \Pi_i$ qui maximise $g(i, \Pi_i \cup Z)$
    $P_{new} \leftarrow g(i, \Pi_i \cup Z)$
    si $P_{new} > P_{old}$ **alors**
      $P_{old} \leftarrow P_{new}$
      $\Pi_i \leftarrow \Pi_i \cup Z$
    sinon
      $Continuer \leftarrow FAUX$
    finsi
  fin tantque
fin pour
Retourner Π.

L'algorithme $K2$ nécessite un ordre total *à priori* sur les sommets afin de s'assurer que le graphe résultant soit sans cycle. En partant du graphe vide, l'algorithme tente itérativement d'ajouter à chaque sommet le parent qui augmente le plus la probabilité de

la structure résultante, exprimée par la fonction g. $K2$ s'arrête lorsque l'ajout de parent ne permet plus d'améliorer la probabilité.

$$g(i, \Pi_i) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} (N_{ijk}!)$$

où :

- Π_i : ensemble des parents du sommet x_i ;
- ϕ_i : liste de toutes les instanciations possibles de x_i dans D ;
- $q_i = |\phi_i|$;
- r_i : nombre de modalités de l'attribut x_i ;
- α_{ijk} : nombre d'exemples de D pour lesquels l'attribut x_i est instancié à sa k^{me} valeur, et ses parents à leur j^{eme} valeur dans ϕ_i ;
- $N_{ij} = \sum_{k=1}^{r_i} (\alpha_{ijk})$, c'est-à-dire le nombre d'exemples de D pour lesquels les parents de x_i sont instanciés à leur j^{eme} valeur dans ϕ_i.

Cet algorithme emploie une heuristique gloutonne, c'est-à-dire qu'il prend à chaque itération une décision optimale locale, dans l'espoir de converger vers l'optimum global. Ceci n'est pas le cas généralement. De plus, la contrainte d'ordonnancement des sommets restreint la recherche à une faible partie des graphes orientés acycliques.

2.3.3 Méta-heuristiques

Afin d'explorer un espace de recherche plus vaste et multimodal, les méta-heuristiques emploient une recherche stochastique, en procédent par échantillonage d'une fonction objectif. Ces algorithmes permettent de formuler un compromis entre l'*exploration* la plus vaste possible de l'espace de recherche et l'exploitation de résultats approchés par le biais d'heuristiques. Au deux extrèmes, l'exploration pure revient à une recherche aléatoire alors que l'exploitation pure serait une recherche locale. Parmi cette famille d'algorithmes, je m'intéresserai aux algorithmes génétiques.

2.4 Applications

De part leur aspect de « boîte blanche », les réseaux bayésiens offrent une bonne compréhensibilité et sont utilisés dans de nombreux domaines d'application. Dans le domaine médical, la nature probabiliste des modèles permet aux experts d'exprimer des hypothèses dans leur mode de raisonnement et de confronter les modèles à leur expérience [NFS04]. Un avantage important est la capacité à gérer les données manquantes avec succès, ce qui est essentiel dans les applications réelles.

3 Algorithmique génétique

3.1 Principe général

Les algorithmes génétiques sont des méthodes de recherche stochastique basées sur des abstractions des processus d'évolution naturelle. Divers ouvrages ont été consacrés aux algorithmes génétiques : [Mit96] constitue une bonne introduction et [Fre02] traite des algorithmes génétiques pour la fouille de donnée et l'apprentissage de règles de décision. L'algorithme 2 ci-dessous décrit une version générique d'un algorithme génétique.

Algorithme 2 Pseudo-code d'un algorithme génétique abstrait

Entrées: Fonction d'évaluation f, nombre d'itérations maximum n

Sorties: Meilleur individu trouvé

```
Générer aléatoirement la population initiale
i ← 0
tantque i < n et Condition d'arrêt non satisfaite faire
    Évaluer chaque individu de la population selon f
    Sélectionner les parents dans la population
    Produire les enfants des parents sélectionnés par croisement
    Muter les individus
    Étendre la population en y ajoutant les enfants
    Réduire la population
    i ← i + 1
fin tantque
Retourner le meilleur individu trouvé.
```

Les idées principales de ce paradigme sont les suivantes : tout d'abord, un algorithme maintient une population d'individus, chacun d'eux représentant une solution à un problème donné. Chaque individu est évalué selon une *fonction objectif*, qui mesure sa qualité de résolution du problème. Les individus évoluent vers de meilleurs solutions par une procédure basée sur la sélection naturelle, c'est-à-dire la survie et la reproduction des mieux adaptés. La reproduction se fait par le biais d'opérateurs génétiques de recombinaison (ou croisement) et de mutation. Plus l'évaluation d'un individu est bonne, plus grande est sa chance que son « matériel génétique » soit transmis aux générations futures d'individus.

De façon informelle, l'opérateur de croisement recombine le matériel génétique des individus, alors que l'opérateur de mutation change la valeur d'un « gène » (plus petit

constituant du bagage génétique d'un individu) pour une valeur aléatoire. Ces deux opérateurs stochastiques sont appliqués avec une probabilité pré-définie. La population évolue jusqu'à satisfaction d'un critère d'arrêt : présence d'un individu satisfaisant ou nombre de générations fixé.

Les algorithmes génétiques sont des méthodes de recherche très flexibles. Elles permettent de résoudre toute sorte de problème, en choisisant une représentation des individus, des opérateurs et une fonction d'évaluation adéquates. Ces choix sont fortement dépendants du problème.

En comparaison avec des méthodes de recherche locale, procédant par opérations déterministes qui modifient une petite partie d'une solution candidate, la recombinaison permet d'en modifier une grande part (en moyenne la moitié). L'hypothèse principale est que la connaissance d'heuristique très spécifique n'est pas nécessaire, mais que de la recombinaison et de la sélection émergent une adaptation naturelle au problème à résoudre.

3.2 Problématique

L'implémentation d'un algorithme génétique est spécifique au problème à résoudre. Les considérations principales sont les suivantes [Bäc00] :

- **Choix du codage des hypothèses** : un codage approprié doit être choisi pour représenter les modèles.
- **Choix de la fonction objectif** : la fonction d'évaluation des hypothèses doit être choisie avec soin, vu que c'est elle qui détermine ce qui sera optimisé par l'algorithme génétique. Il est important que cette fonction ait un grain suffisamment fin, pour pouvoir sélectionner comparativement les meilleurs individus ;
- **Choix de la méthode de sélection** : les trois méthodes principales sont la sélection par *roulette proportionnelle*, par *rang* et par *tournoi*. La roulette assigne une chance d'être sélectionné à chaque individu proportionnellement à son évaluation. Elle est donc biaisée en faveur des individus les plus adaptés. La sélection par rang consiste à trier les individus par ordre d'évaluation, puis sélectionner les meilleurs. En comparaison à la sélection par roulette, la distribution des évaluations est symétrique et il y a donc moins de chance qu'un « super-individu » envahisse la population et conduise à une convergence prématurée. La sélection par tournoi tire au hasard des ensembles d'individus dans la population, et sélectionne dans ces sous-ensemble les individus les plus adaptés. Cette méthode est moins coûteuse en temps de calcul ;

- **Choix des opérateurs génétiques** : le croisement et la mutation doivent être adaptés au type de représentation choisie ;
- **Choix des méta-paramètres** : taille population, probabilités d'opération, critère d'arrêt...

3.3 Algorithme génétique et programmation évolutionnaire

Dans les approches évolutionnaires de la recherche stochastique, deux paradigmes principaux cohabitent.

D'une part, les *algorithmes génétiques* sont la famille d'algorithmes les plus proches de l'évolution naturelle. Ils font usage du croisement comme méthode de recherche principale, alors que la mutation est appliquée avec une probabilité très faible. Les individus sont le plus souvent représentés par des chaînes binaires.

D'autre part, la *programmation évolutionnaire* est une approche plus libérale de ce type d'algorithmes. Elle fait usage de tout type de représentation (listes de réels, structures complexes), et l'opération de recombinaison est souvent délaissée au profit d'une mutation, s'approchant plus de la recherche locale avec heuristique. Ces algorithmes introduisent plus de connaissances sur la résolution du problème et sont parfois appelés *algorithmes hybrides*.

4 Algorithmes génétiques pour l'identification structurelle

4.1 Représentation

Larrañaga ([LP94], puis [LP96] pour une évaluation des paramètres) a proposé le premier une recherche sur l'ensemble des graphes orientés acycliques à l'aide d'algorithmes génétiques. Il propose que les individus soient codés sous forme d'une chaîne binaire représentant leur *matrice de connectivité* (transposée de la matrice d'adjacence). Afin que les opérateurs de croisement et de mutations soient fermés (c'est-à-dire qu'ils produisent des représentations de modèles valides), il restreint l'ensemble des modèles à ceux dont les sommets sont totalement ordonnés.

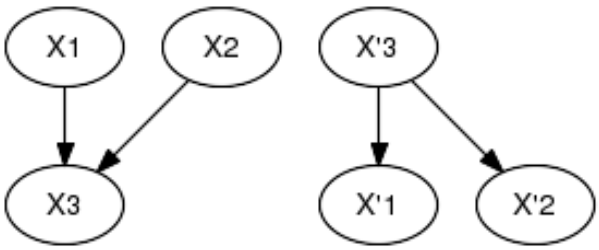


FIG. 2 – *Structures parentes avant croisement*

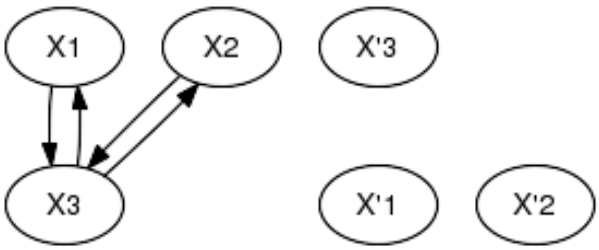


FIG. 3 – *Structure illégale produite par croisement*

Les figures 2 et 3 illustrent des structures illégales qui peuvent être produites lors du croisement. La figure 2 représente deux individus avant croisement. Après croisement (*cf.* figure 3), le premier modèle hérite de deux arcs supplémentaires, ce qui produit des cycles et rend donc ce modèle non valide.

Une solution est de rejeter le modèle non valide, ou encore d'implémenter un opérateur qui corrige le modèle non valide en supprimant les arcs superflus.

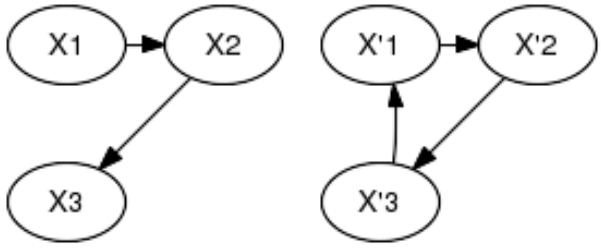


FIG. 4 – *Structure illégale produite par mutation*

Le même problème peut survenir lors d'une mutation (*cf.* figure 4), résultant en une strucure non valide.

4.2 Critère d'évaluation

Le choix de la fonction objectif est guidé par le désir d'avoir un modèle qui à la fois synthétise correctement les données et possède de bonnes performances explicatives et prédictives. Ce choix exprime le compromis *biais-variance*. En effet, l'erreur du modèle provient de deux sources : d'une part le biais induit par le choix de la structure et son adéquation au phénomène représenté, et d'autre part la variance dans le choix des paramètres. Ainsi, un réseau bayésien naïf aura une variance faible dans le choix des paramètres mais un fort biais lié à l'inadéquation de sa structure.

La longueur de description minimale (MDL) [Ris89, LB94] permet de formuler ce compromis. Cette mesure possède deux termes : le premier dénote la complexité du modèle et le second la complexité des données une fois codées par ce modèle.

La mesure MDL possède l'avantage d'être décomposable en somme de mesure locales en chaque sommet, et est donc pratique à évaluer.

4.3 Hybridation

Afin d'améliorer les performances des algorithmes de recherches, de nombreuses méthodes d'hybridation ont été apportées. Larrañaga [LMPK95] propose d'hybrider la mutation en recherchant parmi les parents d'un sommet le meilleur sous-ensemble de sommets. Dans [LEL+99], il effectue une revue des différentes méthodes d'apprentissage de structures.

Van Dijk [vDvdGT03, vDTvdG03, vDT04] effectue tout d'abord une analyse des dépendances d'ordre 0 et 1, afin d'obtenir de bonnes conditions initiales, suivie d'une recherche par algorithmes génétiques.

Wong [WLL99, WL04, WLL04] utilise une approche de *programmation évolutionnaire*, dans laquelle la recombinaison n'est pas utilisée, et la mutation est remplacée par une recherche locale stochastique. La mutation peut être aléatoirement une mutation simple (ajout ou suppression d'arc), une inversion d'arc, un déplacement de parents vers un autre sommet, ou encore une mutation avec heuristique utilisant la mesure MDL.

5 Expérimentation

5.1 Méthodologie

Afin de valider l'utilisation d'algorithmes génétiques pour l'apprentissage de réseaux bayésiens, j'ai effectué un travail d'expérimentation. Celui-ci consiste à concevoir et mettre en œuvre deux algorithmes d'apprentissage dans le logiciel de fouille de données Weka, puis à les valider sur deux jeux de données.

5.1.1 Weka

Weka [Rem05] est un logiciel libre de fouille de donnée et d'apprentissage automatique. Il permet d'appliquer toute la chaîne du processus d'extraction de connaissance à partir de données (pré-traitement, classification supervisée et non supervisée, règles d'associations, visualisation...) La disponibilité du code source permet d'implémenter et de tester ses propres algorithmes tout en s'appuyant sur une plate-forme éprouvée et un code objet, et de comparer leurs performances avec l'existant.

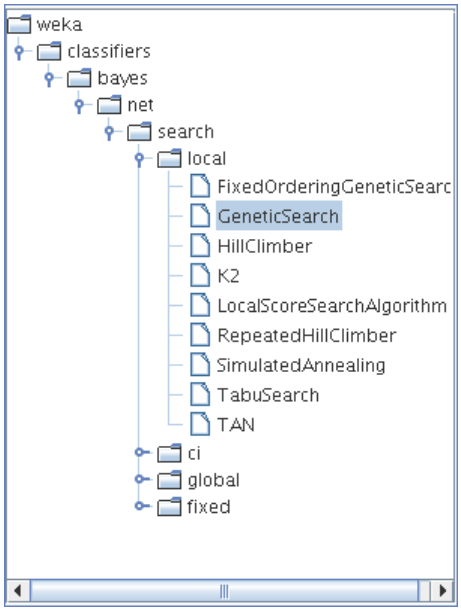


FIG. 5 – *Weka : choix de la méthode de recherche*

Divers algorithmes d'apprentissage de réseaux bayésiens sont implémentés dans Weka (*cf.* figure 5. Ceux-cis sont subdivisés en trois catégories :

- **score local** : l'apprentissage se fait par recherche et optimisation d'un score qui peut être décomposé comme somme de mesures locale en chaque sommet du graphe ;
- **tests d'indépendance conditionnelle** ;
- **score global** : les modèles candidats sont évalués selon leurs performances globales, telles que la précision d'une tâche de classification.

5.1.2 Implémentation

La réalisation s'effectue sur deux algorithmes d'apprentissage : *GeneticSearch* et *FixedOrderingGeneticSearch*.

L'algorithme *GeneticSearch* (*cf.* figure 6) effectue une recherche sur l'ensemble des graphes orientés acycliques, alors que l'algorithme *FixedOrderingGeneticSearch* travaille uniquement sur l'ensemble des modèles dont les sommets sont totalement ordonnés. L'ordre des sommets spécifié par celui des attributs de la base d'apprentissage.

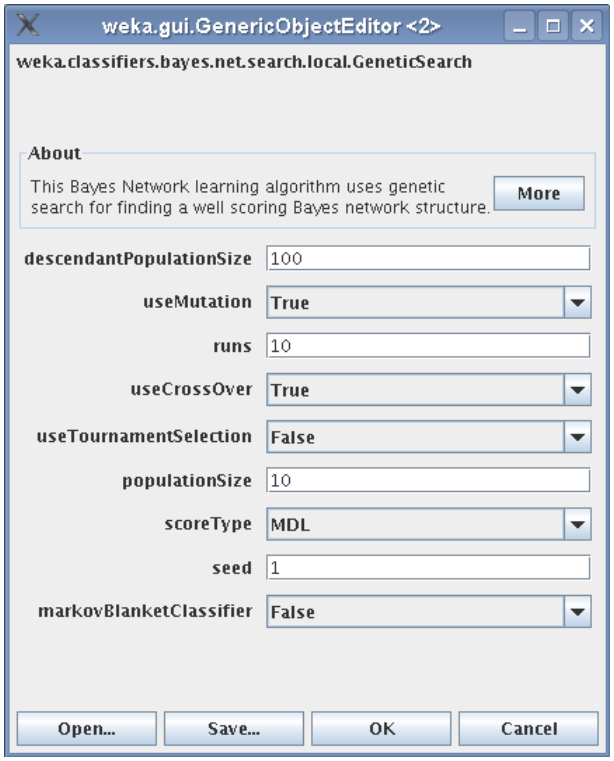


FIG. 6 – *Weka : choix des paramètres de l'algorithme GeneticSearch*

5.2 Évaluation

L'apprentissage de structure de réseau bayésien étant une tache de classification non supervisée, il est difficile d'évaluer les performances des algorithmes et de les comparer entre eux. En effet, la performance d'un algorithme est lié à la compréhensibilité des modèles qu'il produit et à sa capacité à synthétiser les données.

Un approche est de considérer réseau bayésien défini par un expert. Celle-ci définit une distribution jointe de probabilité, à partir des données d'apprentissage peuvent être générées par simulation stochastique. L'objectif de l'algorithme est de reconstituer le réseau qui a le plus probablement généré les données d'apprentissage, à savoir le graphe d'origine.

5.2.1 Réseau Asia

Le réseau Asia, introduit par Lauritzen et Spiegelhalter [LS90], est un graphe fictif illustrant des connaissances médicales. Ce graphe (*cf.* figure 7) est constitué de 8 sommets et 8 arcs. Chaque variable est binaire. Une base d'apprentissage de 10000 exemples a été générée par échantillonage logique [Hen86] à partir du graphe expert.

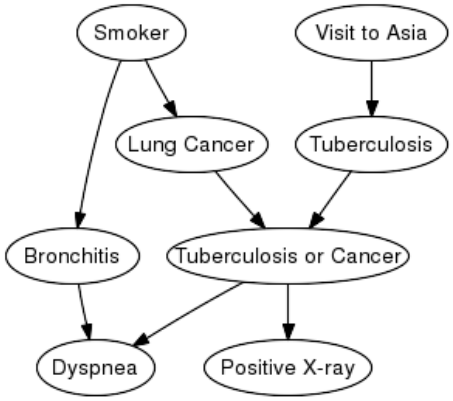


FIG. 7 – *Réseau Asia*

5.2.2 Réseau ALARM

Le réseau ALARM, introduit par Beinlich *et al.* [BSCC89], est un réseau réel créé par des experts médicaux pour surveiller les patients en soins intensifs. ALARM est largement utilisé comme banc d'essai pour comparer les performances des différents algorithmes d'apprentissage. Ce graphe (*cf.* figure 8) comprend 37 sommets et 46 arcs. Les variables

sont discrètes, mais pas nécessairement binaires. Une base de 10000 exemples a également été générée à partir du graphe expert et a servi d'ensemble d'apprentissage.

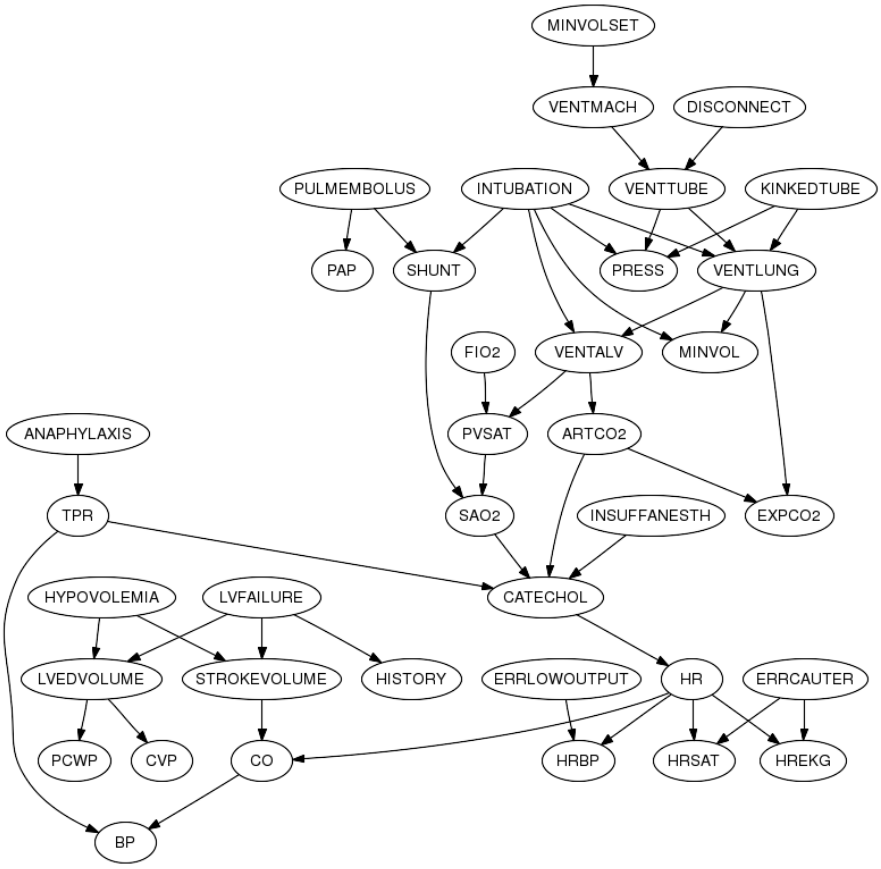


FIG. 8 – *Réseau ALARM*

5.3 Résultats

Trois expériences ont été menées et sont synthétisées dans le tableau 2. Les algorithmes génétiques étant non-déterministes, chaque expérience a été lancée 20 fois avec une initialisation différente du générateur de nombres pseudo-aléatoires (la graine employée est le rang de l'exécution). Pour chacune des expériences, la fonction objectif employée est une mesure de description minimale (MDL). La taille de la population initiale est de 50 individus. La taille de la population des descendants (avant réduction) est de 100 individus.

Les opérateurs de croisement et de mutation sont appliqués comme suit. Un individu est tiré (avec remise) de la population des parents et a une probabilité de croisement de 0.5. Si un croisement est effectué, un autre individu de la population des parents est tiré (avec remise) et le croisement est effectué. Sinon, l'individu a une probabilité de mutation de 0.5. Si l'individu ne subit ni croisement, ni mutation, alors il est transféré tel quel à la population des descendants. Enfin, la réduction consiste en une sélection élitiste par rang, c'est-à-dire que les individus de la population des descendants sont classés par score décroissant et que les 50 premiers individus sont conservés pour la génération suivante.

Expérience	Exécutions	Générations	Réseau	Algorithme
1	20	100	*ASIA*	*FixedOrderingGeneticSearch*
2	20	100	*ASIA*	*GeneticSearch*
3	20	15000	*ALARM*	*FixedOrderingGeneticSearch*

TAB. 2 – Expérimentations

5.3.1 Expérience 1 : réseau Asia, sommets ordonnés

Pour cette première expérience, l'espace de recherche est celui des graphes orientés acycliques dont les sommets sont totalement ordonnés. L'ordre utilisé est celui du réseau expert d'origine, sinon celui-ci, n'appartenant pas à l'espace d'hypothèses, n'aurait pu être reconstitué.

L'expérience montre que sur le réseau factice *Asia*, l'algorithme converge très rapidement : en 30 itérations, chacune des 20 exécutions a réussi à trouver l'optimum de la fonction objectif et à reconstituer le graphe d'origine.

L'aspect « continu »des courbes d'évaluation et l'absence de plateaux suggère que l'algorithme de recherche tire correctement parti de ses capacité d'exploitation de solutions sous-optimale afin de converger vers l'optimum global dans le cas présent d'une fonction d'objectif peu multimodale.

5.3.2 Expérience 2 : réseau Asia, sommets non ordonnés

Pour cette seconde expérience, la recherche s'effectue dans l'espace plus vaste des graphes orientés acycliques.

Cette expérience montre une courbe similaire à l'expérience précédente, mais possédant une vitesse de convergence sensiblement plus lente. En 50 itérations environ, la plupart des exécutions ont convergé vers l'optimum global de la fonction objectif. Pour

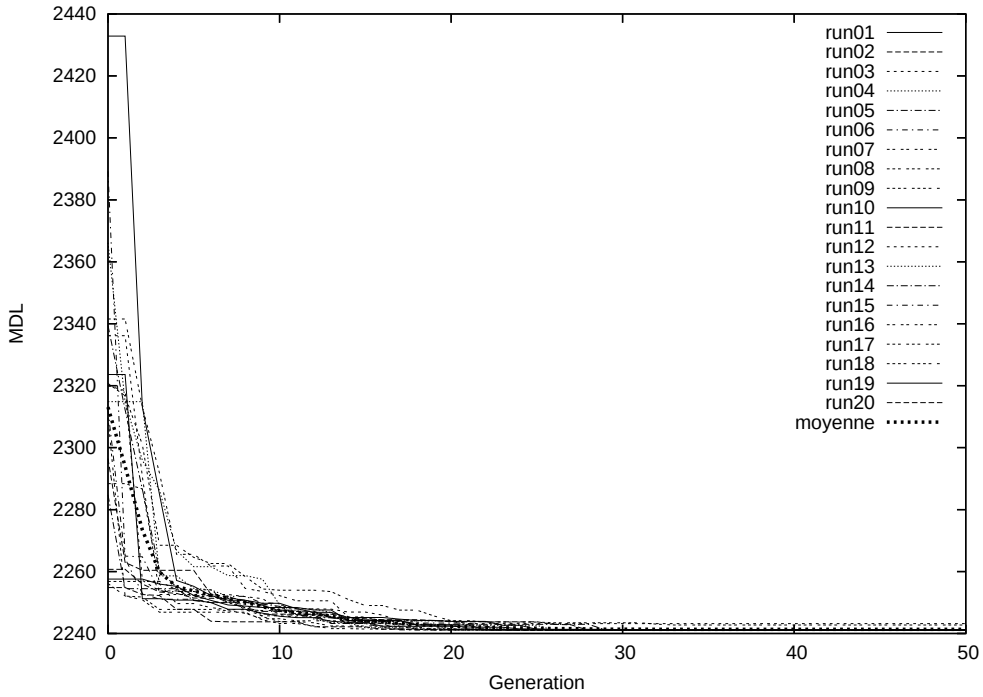


FIG. 9 – *Exp. 1 : réseau Asia, sommets ordonnés*

quelques exécutions, l'algorithme de recherche n'a pas permis de reconstituer le graphe d'origine. Plus précisément, il subsiste en effet 1 arc superflu ou manquant.

Globalement, cet algorithme a comme son prédecesseur un bon comportement sur ce problème et tire parti correctement de ses capacités de recombinaison de solutions partielles.

5.3.3 Expérience 3 : réseau ALARM, sommets ordonnés

La troisième expérience consiste en l'apprentissage du réseau ALARM sur l'espace des graphes orientés acycliques dont les sommets sont totalement ordonnés. La complexité de ce réseau est d'ordre très supérieure aux expériences précédentes, le graphe possédant 37 sommets au lieu de 8.

Le graphique montre qu'après avec fortement décru avec une courbe similaire aux

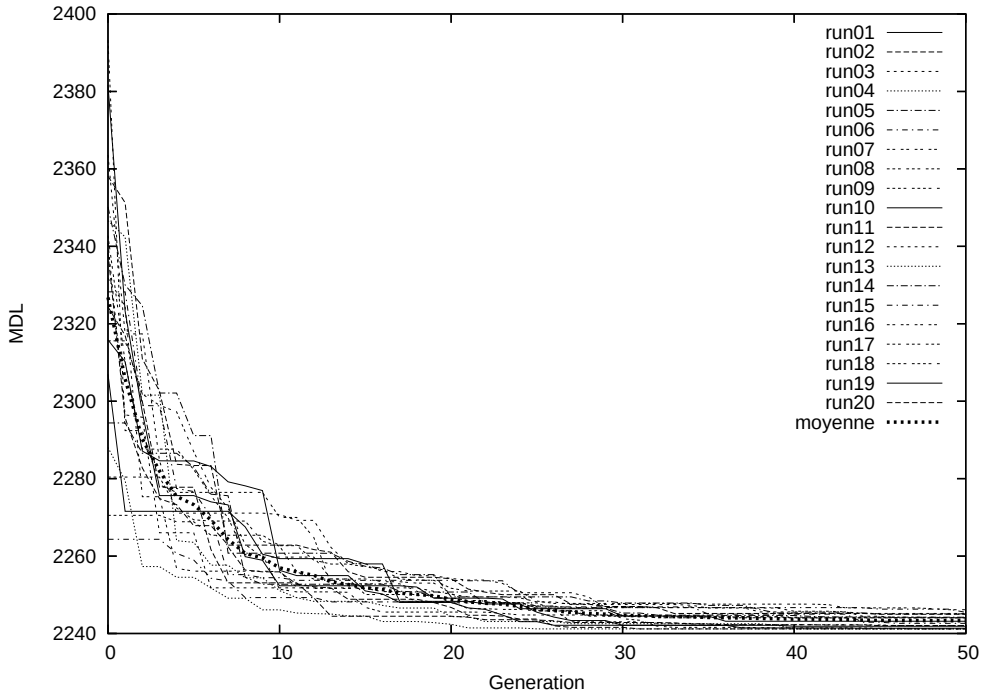


FIG. 10 – *Exp. 2 : réseau Asia, sommets non ordonnés*

tests précédents, l'algorithme tend à se stabiliser pendant de nombreuses générations (parfois plus de 1000). Cette convergence « par palier »suggère que l'algorithme s'est stabilisé sur un optimum local pendant un grand laps de temps, avant de trouver une mutation lui permettant de s'éloigner du sous-optimum pour explorer une autre région de l'espace des hypothèses, et l'exploiter jusqu'à convergence vers le sous-optimum suivant. Le compromis exploration / exploitation est donc bien visible, et la courbe se comporte asymptotiquement comme une exploration aléatoire avec des phases de recherche locale.

5.3.4 Diversité des individus

Afin de vérifier cette hypothèse, j'ai mesuré la diversité des individus au sein d'une population à un temps donné. La mesure de diversité est calculée comme le rapport du nombre d'individus distincts sur le nombre total d'individu. Ainsi, une diversité de 1

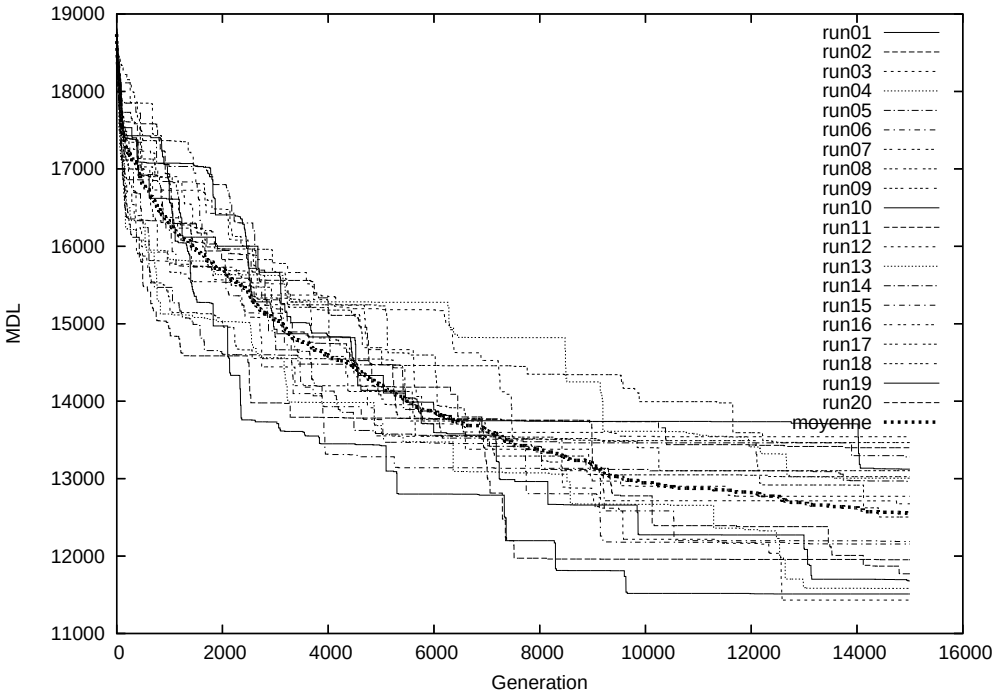


FIG. 11 – *Exp. 3 : réseau ALARM, sommets ordonnés*

correspond à l'initialisation de l'algorithme (tous les individus ont été générés aléatoirement), alors qu'une diversité de 0 correspond à la convergence de l'algorithme (un modèle optimal a été trouvé et a envahi toute la population).

$$diversite = \frac{nombre\ d'individus\ distincts}{nombre\ d'individus}$$

La figure 12 correspond à une seule exécution de l'algorithme précédent (avec la graine 52) et permet de visualiser simultanément la valeur de fonction objectif et la diversité des individus. On constate que l'hypothèse est vérifiée. Les plateaux de la fonction objectifs correspondent à une diversité nulle : tous les individus sont alors identiques. Lorsqu'une mutation pertinente se produit, la fonction d'objectif décroit pendant un certain nombre de générations alors qu'il se produit simultanément un pic de la fonction de diversité.

On peut donc dire, si l'on suppose que les phases d'explorations et d'exploitation sont

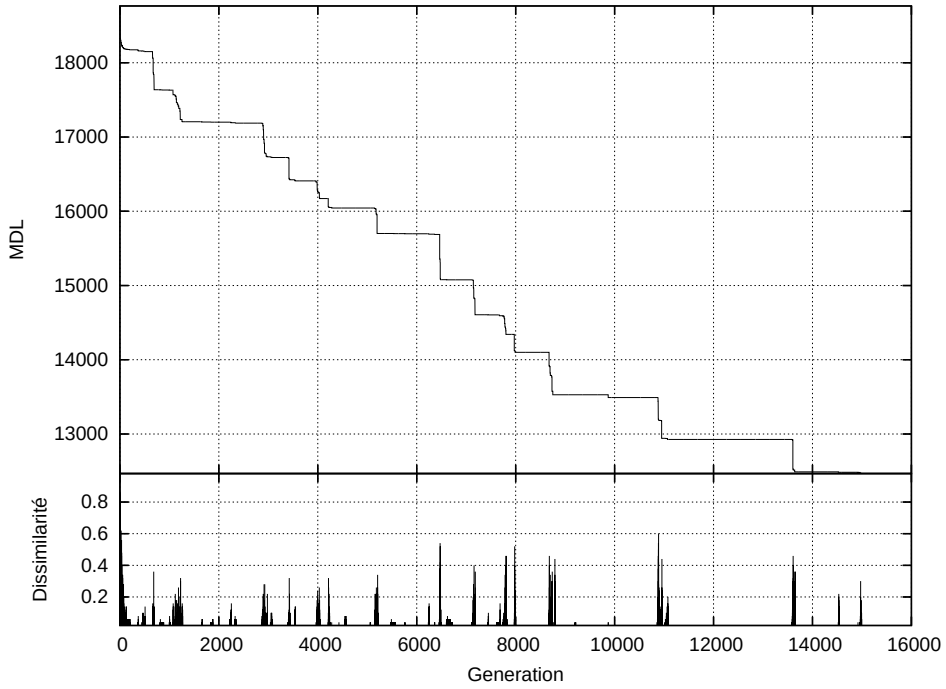


FIG. 12 – *Exp. 3 : Diversité des individus (réseau ALARM, sommets ordonnés)*

complètement disjointes, que sur la figure 12, les pics de la fonction diversité correspondent grossièrement à la phase d'exploitation de l'algorithme de recherche, alors que les plateaux montrent la phase d'exploration.

6 Conclusion

Ce stage a permi d'étudier et de réaliser différents algorithmes d'apprentissage automatique de réseaux bayésiens. Les programmes résultants sont tout à fait utilisables dans des applications réelles et dans des temps raisonnables. Les algorithmes permettent de formuler un compromis exploration contre exploitation en fonction de la difficulté des données analysées. Cependant, il est toujours possible d'améliorer les performances en vue d'un problème particulier. La démarche employée ici était la plus générale possible, et les approches hybrides de la programmation évolutionnaire semblent donner de bons résultats pour spécialiser les algorithmes génétiques si besoin est.

Références

[BSCC89] I.A. Beinlich, H.J. Suermondt, R.M. Chavez, and F. Cooper. The ALARM Monitoring System : A Case Study with Two Probabilistic Inference Techniques for Belief Networks. In *Second European Conf. Artificial Intelligence in Medicine*, pages 247–256, 1989.

[Bäc00] Thomas Bäck. *Evolutionary Computation 1 : Basic Algorithms and Operators*. Taylor & Francis, 2000.

[CH92] Gregory F. Cooper and Edward Herskovits. A bayesian method for the induction of probabilistic networks from data. *Mach. Learn.*, 9(4) :309–347, 1992.

[Fre02] A.A. Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer, 2002.

[Hen86] Max Henrion. Propagating uncertainty in bayesian networks by probabilistic logic sampling. In *Proceedings of the 2nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-86)*, pages 0–0, New York, NY, 1986. Elsevier Science.

[LB94] Wai Lam and Fahiem Bacchus. Learning Bayesian Belief Networks : An Approach Based on the MDL Principle. *Computational Intelligence*, 10 :269–294, 1994.

[LEL+99] P. Larrañaga, R. Etxeberrîa, J.A. Lozano, B. Sierra, I. Inza, and J.M. Peña. A review of the cooperation between evolutionary computation and probabilistic graphical models. In *Proceedings of the Second Symposium on Artificial Intelligence, CIMAF 99, Havana, Cuba*, pages 314–324, 1999.

[LMPK95] Pedro Larrañaga, R. Murga, M. Poza, and C. Kuijpers. Structure learning of Bayesian networks by hybrid genetic algorithms, 1995.

[LP94] Pedro Larrañaga and M. Poza. Structure learning of Bayesian networks. In E. Diday, editor, *Studies in Classification, Data Analysis, and Knowledge Organization*, pages 300–306. Springer-Verlag, 1994.

[LP96] P. Larrañaga and M. Poza. Structure Learning of Bayesian Networks by Genetic Algorithms : A Performance Analysis of Control Parameters. *IEEE Journal on Pattern Analysis and Machine Intelligence*, 18(9) :912–926, 1996.

[LS90] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Readings in uncertain reasoning*, pages 415–448, 1990.

[Mit96] Melanie Mitchell. *An introduction to genetic algorithms.* The MIT Press, 1996.

[Nea03] R.E. Neapolitan. *Learning Bayesian Networks.* Pearson Prentice Hall, 2003.

[NFS04] Andy Novobilski, Francis Fesmire, and David Sonnemaker. Mining bayesian networks to forecast adverse outcomes related to acute coronary syndrome. In *FLAIRS Conference*, 2004.

[Rem05] Remco Bouckaert Remco. Bayesian network classifiers in weka, 2005.

[Ris89] Jorma Rissanen. *Stochastic Complexity in Statistical Inquiry.* World Scientific Pub Co Inc, 1989.

[vDT04] Steven van Dijk and Dirk Thierens. On the Use of a Non-redundant Encoding for Learning Bayesian Networks from Data with a GA. In *PPSN*, pages 141–150, 2004.

[vDTvdG03] Steven van Dijk, Dirk Thierens, and Linda C. van der Gaag. Building a GA from Design Principles for Learning Bayesian Networks. In *GECCO*, pages 886–897, 2003.

[vDvdGT03] Steven van Dijk, Linda C. van der Gaag, and Dirk Thierens. A Skeleton-Based Approach to Learning Bayesian Networks from Data. In *PKDD*, pages 132–143, 2003.

[WL04] Man Leung Wong and Kwong-Sak Leung. An efficient data mining method for learning Bayesian networks using an evolutionary algorithm-based hybrid approach. *IEEE Trans. Evolutionary Computation*, 8(4) :378–404, 2004.

[WLL99] Man Leung Wong, Wai Lam, and Kwong-Sak Leung. Using Evolutionary Programming and Minimum Description Length Principle for Data Mining of Bayesian Networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(2) :174–178, 1999.

[WLL04] Man Leung Wong, Shing Yan Lee, and Kwong-Sak Leung. Data mining of Bayesian networks using cooperative coevolution. *Decision Support Systems*, 38(3) :451–472, 2004.