Une unification des algorithmes d'inférence de Pearl et de Jensen

Olfa Ben Naceur-Mourali* — **Christophe Gonzales****

* *Institut Supérieur de Gestion*
Le Bardo 2000, Tunisie
bnaceur.oltar@planet.tn

** *LIP6 – université Paris 6 – pôle IA*
8, rue du capitaine Scott
75252 Cedex 05 Paris, France
Christophe.Gonzales@lip6.fr

RÉSUMÉ. *Les réseaux bayésiens sont un des outils les plus populaires de la communauté Intelligence Artificielle pour gérer des incertitudes. Ils allient en effet un aspect graphique très expressif et des mécanismes d'inférence de nouvelles informations très efficaces. Deux catégories d'algorithmes existent à cet effet : les méthodes non orientées travaillant sur des structures secondaires telle que celle de Jensen et les méthodes comme celle de Pearl, travaillant sur le réseau bayésien d'origine. Il est communément admis que les premières sont nettement plus efficaces que les secondes. Nous présentons dans cet article un algorithme permettant d'obtenir une structure secondaire orientée très liée à celle de Jensen et nous montrons que des calculs similaires à ceux de Pearl menés dans cette nouvelle structure atteignent des performances identiques à celles de l'algorithme de Jensen.*

ABSTRACT. *By combining a very expressive graph and efficient inference mechanisms, Bayesian networks have become increasingly popular among the Artificial Intelligence community for dealing with uncertainty. Methods for propagating evidence in the network can be divided into two classes: i) those that, like Pearl, use the Bayesian network structure to conduct the inference process; and ii) those that, like Jensen, create undirected secondary structures and perform computations in these new structures. It is commonly thought that the latter are much more efficient than the former. In this paper, we present an algorithm that constructs a new directed secondary structure, much related to Jensen's one, and we show that Pearl's algorithm whithin this new structure is as efficient as Jensen's algorithm.*

MOTS-CLÉS : *réseaux bayésiens, inférence, propagation, triangulation, arbres de jonction.*

KEYWORDS: *Bayesian networks, inference, propagation, triangulation, junction trees.*

1. Introduction

Les réseaux bayésiens sont nés dans les années 80 d'un besoin de gérer efficacement les incertitudes dans les systèmes experts à base de règles. En effet, les règles du type « si X est vrai, alors Y l'est aussi » présentaient deux inconvénients majeurs : tout d'abord, dans les situations pratiques, leur pouvoir déductif était limité dans le sens où une connaissance imparfaite de X ne permettait pas d'inférer Y, faute de pouvoir activer la règle. Ensuite, elles nécessitaient une gestion complexe des exceptions, ne serait-ce que pour assurer la cohérence de la base. Pour pallier ces inconvénients, [SHO 76] introduisit avec MYCIN les facteurs de certitude, avec des règles dans lesquelles, lorsque X est vrai, Y l'est aussi mais seulement avec un certain degré de certitude. Malheureusement, les facteurs de certitude avaient, eux aussi, de mauvaises propriétés, par exemple le traitement incorrect des informations corrélées ou bien certaines abductions erronées. C'est ce qui amena Judea Pearl à proposer un nouveau modèle de règles fondé sur les probabilités : les réseaux bayésiens [PEA 88].

Rapidement, ceux-ci sont devenus l'un des outils les plus populaires de la communauté Intelligence Artificielle pour gérer les incertitudes. Leur succès est certainement dû au fait qu'ils allient une représentation compacte des probabilités et des mécanismes de calcul efficaces. En particulier, ils permettent d'inférer l'impact d'une nouvelle information sur une loi de probabilité (calcul de probabilités *a posteriori*). De plus, la représentation utilisée, un graphe orienté dont chaque nœud correspond à une variable aléatoire et dont les arcs représentent qualitativement des dépendances entre les variables, permet de visualiser aisément les interactions entre variables. Enfin, la force des dépendances est quantifiée grâce à des probabilités conditionnelles, ce qui permet d'obtenir une loi de probabilité jointe des variables à partir de sources hétérogènes (des experts et/ou des bases de données). Ces avantages ont été exploités dans de nombreux domaines. Citons par exemple le diagnostic de pannes ([BRE 99], [NIE 00]) ou bien la modélisation d'utilisateurs ([CON 97], [MIS 96], [HOR 98]). Voir [JEN 96] ou [BEC 99b] pour des exemples détaillés.

Avec l'amélioration de la puissance de calcul des ordinateurs, un besoin s'est fait sentir de « coller » de plus près à la réalité, avec pour résultat de complexifier de plus en plus la structure des réseaux. Cette sophistication a bien entendu dégradé les performances de l'algorithme d'inférence proposé par Pearl et a favorisé la recherche de nouveaux algorithmes plus performants. En 1988, [LAU 88] a proposé une méthode de calcul très différente de celle de Pearl. Elle fut par la suite améliorée par Jensen [JEN 90]. Depuis, d'autres algorithmes sont apparus, qui sont tous fondés sur un principe similaire ([SHA 90] et [MAD 99]).

Lorsque le réseau est sans cycles, il est communément admis que tous les algorithmes d'inférence ont des performances similaires. En revanche, lorsque le réseau contient des cycles, il est indéniable que la nouvelle vague d'algorithmes ([JEN 90], [SHA 90], etc.) est supérieure à celui de Pearl et à ses variantes ([PEO 91]). Cette assertion fut démontrée dans [SHA 94]. Depuis, quelques tentatives d'améliorations de l'algorithme de Pearl ont été proposées ([DAR 95], [DIE 96] et [FAÕ0]), mais celles-

ci ne rivalisent pas avec les méthodes à la Jensen. Toutefois, les algorithmes à la Pearl présentent un avantage que les algorithmes fondés sur [JEN 90] ne possèdent pas : ils ont à leur disposition des informations (la *d*-séparation) qui leur permettent de détecter les zones du réseau dans lesquelles les calculs d'inférence sont inutiles. Or, en pratique, cela permet une réduction considérable des temps de calcul. C'est pourquoi dans cet article, nous présentons une nouvelle variante de l'algorithme de Pearl qui, tout en conservant cet avantage, obtient une complexité algorithmique comparable à celle de Jensen (sans avoir recours à la *d*-séparation).

Dans la section 2, nous présentons brièvement les réseaux bayésiens et les illustrons sur l'exemple classique de [LAU 88]. La section 3 décrit l'algorithme d'inférence proposé par Jensen. Dans une première phase, celui-ci construit une structure secondaire non orientée appelée *arbre de jonction* et, dans un deuxième temps, il effectue des calculs probabilistes dans cette structure. En nous fondant sur deux remarques concernant l'arbre de jonction, nous motivons dans la section 4 un algorithme transformant le réseau bayésien en un nouveau réseau orienté intimement lié à l'arbre de jonction. Dans la section 5, une légère variante de l'algorithme de [FAÕ0] est présentée qui, appliquée à ce nouveau réseau, atteint une performance égale à celle de l'algorithme de Jensen. Les démonstrations sont fournies en appendice.

2. Qu'est-ce qu'un réseau bayésien ?

Définition 1 *Un réseau bayésien est un triplet $(\mathcal{V}, \mathcal{A}, \mathcal{P})$ représentant les connaissances que l'on possède d'un domaine d'étude, et dans lequel :*

1) $\mathcal{V} = \{X_1, \ldots, X_n\}$ est un ensemble de nœuds/variables aléatoires, que nous supposerons discrètes dans la suite de cet article ;

2) $\mathcal{A} \subseteq \mathcal{V} \times \mathcal{V}$ est un ensemble d'arcs tel que le graphe $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ ne contient pas de circuit[1]*. On dit alors que le graphe est un* DAG[2] *;*

3) $\mathcal{P}$ est l'ensemble des lois de probabilité de chaque nœud de $\mathcal{V}$ conditionnellement à toutes les configurations de ses parents dans le graphe $\mathcal{G}$. Nous écrirons symboliquement : $\mathcal{P} = \{P(X|Pa(X)) : X \in \mathcal{V}\}$, où $Pa(X)$ est l'ensemble des parents de X.

Par exemple, dans le domaine médical, les nœuds/variables aléatoires correspondent aux symptômes et aux maladies, les arcs représentent les liens entre ces symptômes et ces maladies, et les probabilités quantifient la puissance de ces liens, c'est-à-dire la corrélation entre les variables.

---

1. Un circuit est un ensemble de nœuds $\{X_{i_1}, \ldots, X_{i_k}\}$ tel que, pour tout $j \in \{1, \ldots, k-1\}$, il existe un arc de X_{i_j} vers $X_{i_{j+1}}$, et tel que $X_{i_k} = X_{i_1}$.
2. DAG : acronyme de directed acyclic graph.

En théorie des probabilités, il est connu que la formule :

$$P(X_1 = x_1, \ldots, X_n = x_n) = P(X_1 = x_1) \times \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}) \; \forall x_1, \ldots, x_n,$$

que nous écrirons symboliquement :

$$P(X_1, \ldots, X_n) = P(X_1) \times \prod_{i=2}^{n} P(X_i | X_1, \ldots, X_{i-1}), \qquad [1]$$

est valable quelles que soient les relations entre les différentes variables X_i de $\mathcal{V}$[3].

Si l'on possède la connaissance que, pour un indice i donné dans $\{2, \ldots, n\}$, l'ensemble $\{1, \ldots, i-1\}$ peut être partitionné en deux sous-ensembles K_i et L_i tels que $K_i \cap L_i = \emptyset$ et que la variable aléatoire X_i soit indépendante des variables X_l, $l \in L_i$, conditionnellement à $\{X_k : k \in K_i\}$, alors

$$P(X_i | X_1, \ldots, X_{i-1}) = P(X_i | X_k, k \in K_i),$$

et donc, si le graphe $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ est conçu de telle manière que $\mathcal{A} = \{(X_k, X_i) : i \in \{1, \ldots, n\}, k \in K_i\}$, la loi de probabilité jointe sur $\mathcal{V}$ sera égale à :

$$P(\mathcal{V}) = P(X_1, \ldots, X_n) = P(X_1) \times \prod_{i=2}^{n} P(X_i | X_k, k \in K_i) = \prod_{i=1}^{n} P(X_i | \mathit{Pa}(X_i)).$$

Le graphe $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ représente donc simplement des indépendances probabilistes entre les différentes variables. En pratique, il s'avère que les probabilités conditionnelles $P(X_i | X_k, k \in K_i)$ requièrent beaucoup moins d'espace de stockage que celles de l'équation [1]. Elles rendent ainsi possible le stockage sur ordinateur des probabilités de domaines très complexes, là même où la loi de probabilité jointe est si volumineuse qu'elle ne peut raisonnablement être stockée sur aucun ordinateur.

À titre d'illustration, considérons un exemple médical fictif dû à [LAU 88] : « La dyspnée, une maladie respiratoire, peut être engendrée par une tuberculose, un cancer du poumon, une bronchite, par aucune de ces maladies ou bien par plusieurs d'entre elles. Des statistiques ont montré qu'un séjour récent en Asie augmente le risque de tuberculose. Il est bien connu que fumer est un facteur de risque pour le cancer du poumon et la bronchite. Enfin, on sait qu'une radio des poumons ne permet pas de discriminer entre un cancer du poumon et une bronchite, de même, d'ailleurs, que la présence ou l'absence de dyspnée ». De cet exemple particulièrement déprimant, on peut déterminer les variables suivantes : « dyspnée », « tuberculose », « cancer du poumon », « bronchite », « séjour en Asie », « fumer », « radio », toutes ces variables étant booléennes. Les arcs du réseau bayésien peuvent être aisément déduits du texte

3. Cela correspond à une simple application récursive de la formule $P(A, B) = P(A|B)P(B)$.

ci-dessus. En effet, puisque la dyspnée peut être engendrée par une tuberculose, un cancer du poumon et/ou une bronchite, il doit y avoir des arcs entre « dyspnée » et ces trois variables. Il reste maintenant à les orienter. D'une manière générale, on essaye d'orienter des variables causes vers les conséquences car cela permet souvent d'obtenir des réseaux peu denses (petit nombre d'arcs). Cela dit, rien n'interdit d'utiliser l'orientation inverse, la règle de Bayes permettant d'intervertir de manière probabiliste causes et conséquences. Puisqu'un séjour récent en Asie augmente les risques de tuberculose, le graphe doit contenir un arc de « séjour en Asie » vers « tuberculose ». En itérant ce procédé sur l'ensemble du texte, on obtient le réseau de la figure 1.

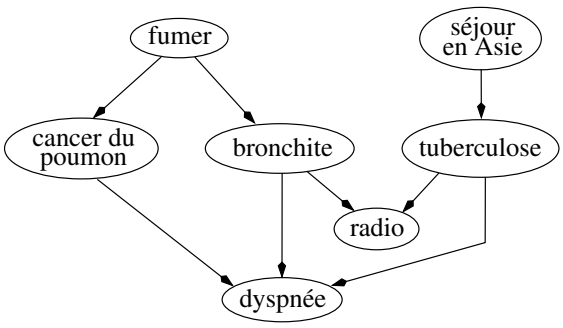


Figure 1. *Le réseau bayésien de la dyspnée*

Une fois le graphe constitué, on stocke dans chaque nœud ses lois de probabilités conditionnelles à toutes les configurations de ses parents dans le graphe. On possède alors une décomposition de la loi jointe de toutes les variables aléatoires et il est donc possible de calculer n'importe quelle probabilité marginale (probabilité qu'une variable prenne telle valeur), jointe (probabilité qu'un groupe de variables prenne tel ensemble de valeurs), conditionnelle (probabilité qu'une variable prenne telle valeur, sachant la valeur d'une autre variable), etc. Ainsi, on peut interroger le réseau de la figure 1 pour savoir quelle est la probabilité qu'un patient quelconque se rendant chez un médecin soit atteint de dyspnée. On dit alors que l'on calcule la probabilité *marginale a priori* de la dyspnée, autrement dit $P(\text{dyspnée})$. Si le patient affirme au médecin être fumeur et n'avoir jamais été en Asie, le réseau peut fournir la probabilité *marginale a posteriori* de la dyspnée, c'est-à-dire $P(\text{dyspnée}|\text{fumeur,non séjour en Asie})$. Intéressons-nous maintenant au calcul de ces probabilités.

3. L'algorithme de Jensen

L'algorithme d'inférence de Jensen, Lauritzen et Olesen, que nous appellerons plus simplement « algorithme de Jensen » par la suite, n'utilise pas directement la structure de graphe du réseau bayésien, mais il emploie une structure secondaire appelée *arbre de jonction*. Celle-ci, bien plus adaptée aux calculs que le réseau bayésien d'origine, permet à Jensen d'être, par là même, beaucoup plus efficace que Pearl, ce dernier

travaillant sur le graphe d'origine. Afin d'unifier ces deux méthodes de calcul, nous allons montrer dans la section suivante comment l'arbre de jonction peut être adapté pour l'algorithme de Pearl. C'est pourquoi, dans cette section, nous allons décrire en détail sa construction.

3.1. *La génération de l'arbre de jonction*

La génération de l'arbre de jonction passe par les trois étapes suivantes :

1) moralisation et suppression des orientations des arcs,

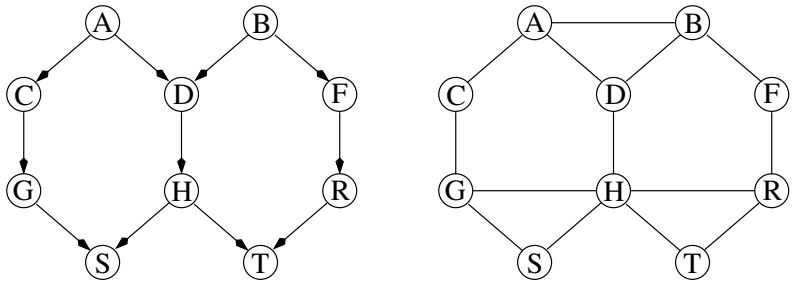
2) triangulation,

3) construction d'un arbre à partir des cliques du graphe triangulé.

La phase de moralisation consiste à reformuler la décomposition de la loi jointe sous forme d'un graphe non orienté. Elle découle de la propriété suivante de l'indépendance conditionnelle : soient X, Y, Z des variables aléatoires ou des groupes de variables, alors X est indépendante de Y conditionnellement à Z si et seulement si les expressions suivantes, qui sont équivalentes, sont vérifiées :

1) $P(X, Y, Z) = P(X|Z)P(Y, Z) = P(X|Z)P(Y|Z)P(Z)$;

2) $P(X, Y, Z)$ est de la forme $a(X, Z)b(Y, Z)$, où a et b sont des fonctions quelconques.



(a) réseau d'origine (b) graphe moralisé

Figure 2. *La phase de moralisation*

Lorsque l'on utilise la première propriété, on obtient une décomposition « orientée » dans le sens où elle fournit des probabilités conditionnelles (les parents des nœuds sont les variables à droite des barres de conditionnement). Ainsi $P(\mathcal{V}) = P(T|H, R)\ P(S|G, H)P(R|F)P(H|D)P(G|C)P(F|B)P(D|A, B)\ P(C|A)\ P(B)\ P(A)$ produit-il le graphe de la figure 2a. En revanche, si l'on applique la propriété 2 à la même loi jointe, on ne peut plus obtenir un graphe orienté, simplement parce que les fonctions a et b de la propriété ne contiennent pas de barre de conditionnement. L'idée consiste alors à relier entre elles toutes les variables dont dépendent ces fonctions. Ainsi, une application de la propriété 2 similaire à celle de la propriété 1, qui

avait abouti à la décomposition de la loi jointe ci-dessus, permet d'obtenir la décomposition suivante :

$$P(\mathcal{V}) = a(T,H,R)b(S,G,H)c(R,F)d(H,D)e(G,C) \\ f(F,B)g(D,A,B)h(C,A)i(B)j(A), \qquad [2]$$

ce qui permet d'en déduire le graphe moralisé de la figure 2b. Pour synthétiser, la moralisation consiste simplement à relier tous les parents d'un même nœud et à supprimer les orientations (les arcs deviennent donc des liens).

Remarquons que les cliques du graphe moralisé correspondent aux ensembles de variables des fonctions a, b, c, etc. (après absorption, telle que celle de $j(A)$ par $h(C,A)$ par exemple, chaque fois que c'est possible). Une clique est un ensemble maximal de nœuds tel qu'il existe un lien reliant tout couple de variables de cet ensemble. L'arbre de jonction est précisément formé à partir de ces cliques :

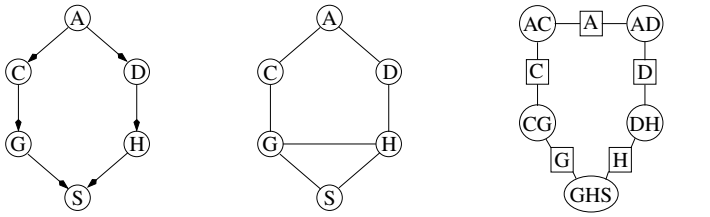
Définition 2 (arbre de jonction) *Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ un graphe non orienté, et soit $\mathcal{C}$ l'ensemble des cliques de $\mathcal{G}$. Un arbre de jonction est un arbre*[4] *dont les nœuds sont les éléments de $\mathcal{C}$ et dont les liens vérifient la propriété d'intersection courante : Si C_1 et C_2 sont des cliques de $\mathcal{C}$ alors, dans toute chaîne reliant C_1 et C_2, toutes les cliques de la chaîne contiennent $C_1 \cap C_2$.*

Malheureusement, du graphe moralisé on ne peut pas systématiquement déduire un arbre de jonction. En effet, comme le montre la figure 3, le graphe de jonction obtenu peut encore contenir des cycles. Dans cette figure, les cliques du graphe de jonction sont représentées par des ellipses et les intersections, que l'on appelle aussi *séparateurs*, par des rectangles. Or, l'algorithme d'inférence de Jensen ne fonctionne que lorsque le graphe de jonction est un arbre. C'est pourquoi, après la moralisation, on effectue une phase de triangulation. En effet, pour un graphe $\mathcal{G}$ donné, il existe un arbre de jonction correspondant si et seulement si $\mathcal{G}$ est triangulé (cf. [COW 99, page 53] pour une démonstration détaillée).

Un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ est triangulé si et seulement si tout cycle de longueur 4 ou plus possède une corde, c'est-à-dire que le cycle contient un couple de nœuds non adjacents dans ce cycle et reliés par un lien dans $\mathcal{E}$. Comme l'a montré [ROS 70], trianguler le graphe moralisé revient à appliquer l'algorithme suivant :

Algorithme 1 (triangulation) *Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ un graphe non orienté, où $\mathcal{V} = \{X_1, \ldots, X_n\}$. Soit σ une permutation de $\{1, \ldots, n\}$. On dit qu'on élimine un nœud X_i de $\mathcal{G}$ lorsque l'on rajoute à $\mathcal{E}$ des liens entre tous les voisins de X_i (X_i et ses voisins*

4. Un arbre est un graphe connexe sans cycle. Un graphe est connexe si tout couple de nœuds est relié par une chaîne. Une chaîne reliant un nœud X à un nœud Y est un ensemble de nœuds $\{X_1, \ldots, X_k\}$ tels que $X_1 = X$, $X_k = Y$, et tel que pour tout $i \in \{1, \ldots, k-1\}$, il existe un lien reliant X_i et X_{i+1}. Enfin, un cycle est une chaîne reliant un nœud à lui-même sans passer deux fois par le même lien.



(a) réseau d'origine (b) graphe moralisé (c) graphe de jonction

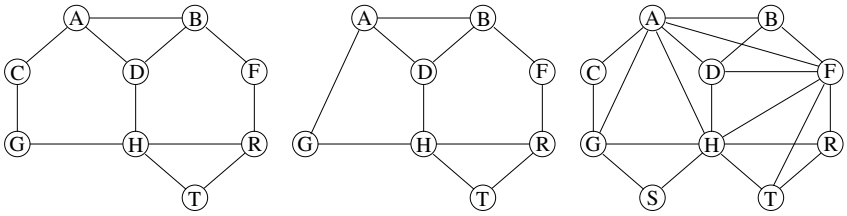
Figure 3. *Graphe moralisé et graphe de jonction*

forment alors une clique) et qu'ensuite on supprime X_i ainsi que ses liens adjacents de $\mathcal{G}$.

Trianguler $\mathcal{G}$ consiste : 1) à éliminer successivement les nœuds $X_{\sigma(1)}, \ldots, X_{\sigma(n)}$, et à créer $\mathcal{E}_T$ l'ensemble des liens ajoutés successivement à $\mathcal{G}$ lors de ces éliminations ; 2) à former le graphe $\mathcal{G}_T = (\mathcal{V}, \mathcal{E} \cup \mathcal{E}_T)$.

Théorème 1 (Rose 70) *Le graphe $\mathcal{G}_T$ obtenu à l'issue de l'algorithme 1 est triangulé.*

Autrement dit, trianguler un graphe revient à appliquer une séquence d'élimination de nœuds. Bien entendu, des séquences différentes vont aboutir à des arbres de jonction différents, et à des complexités différentes dans les calculs de probabilités. Dans cet article, nous ne nous intéresserons pas à chercher la meilleure séquence possible. En effet, nous montrerons que les calculs de notre variante de Pearl rivalisent avec ceux de Jensen, et ce quelle que soit la séquence d'élimination choisie. Cela dit, le lecteur intéressé par les méthodes permettant de trouver de « bonnes » triangulations[5] pourra se reporter à [KJÆ 90], à [BEC 96b] ou bien encore à [SHO 97].



(a) après élimination de S (b) après élimination de C (c) le graphe triangulé

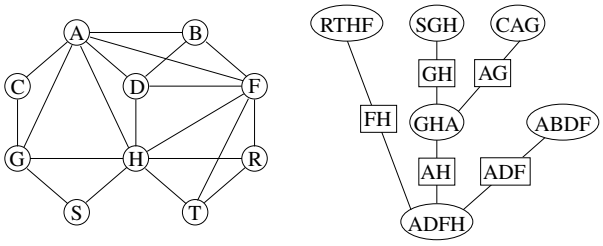
Figure 4. *La phase de triangulation*

À titre d'exemple, reprenons le graphe moralisé de la figure 2b et appliquons la séquence d'élimination S, C, G, R, T, B, D, H, F, A. L'élimination du nœud S ne

5. Trouver la triangulation optimale est un problème NP-difficile comme l'a montré [YAN 81], c'est pourquoi on se contente d'heuristiques.

rajoute aucun lien puisque les voisins de S, les nœuds G et H, sont déjà reliés entre eux. À l'issue de cette élimination, on obtient le graphe de la figure 4a. Lorsque l'on élimine C, on doit rajouter le lien (A, G) car A et G sont les voisins de C et ne sont pas encore reliés. À l'issue de cette élimination on obtient alors le graphe de la figure 4b, et ainsi de suite. Enfin, le graphe triangulé $\mathcal{G}_T$ obtenu en rajoutant au graphe moralisé les liens de triangulation est représenté sur la figure 4c.

L'arbre de jonction est obtenu en prenant l'ensemble des cliques du graphe triangulé $\mathcal{G}_T$, et en les reliant de manière à respecter la propriété de l'intersection courante. Pour cela, le lecteur pourra se reporter à [COW 99, page 60]. Notons que pour un graphe triangulé, il peut exister plusieurs arbres de jonction. Cela dit, comme le montre [JEN 94], tous ces arbres possèdent les mêmes cliques et les mêmes séparateurs. À titre d'illustration, la figure 5 montre un arbre de jonction correspondant au graphe triangulé de la figure 4c.



(a) le graphe triangulé (b) l'arbre de jonction

Figure 5. *L'arbre de jonction*

3.2. *L'inférence dans l'arbre de Jonction*

Une fois l'arbre de jonction construit, Jensen associe à chaque clique et à chaque séparateur un *potentiel*, c'est-à-dire une fonction de l'ensemble des variables de la clique. Celles-ci correspondent, dans l'esprit, aux fonctions a, b, etc., que nous avions obtenues dans l'équation [2]. Nous ne décrirons pas ici l'algorithme permettant d'initialiser ces potentiels et nous laisserons le lecteur se reporter à [JEN 96, pages 73-87]. Simplement, notons qu'à l'issue de cette phase d'initialisation les potentiels contiennent les probabilités jointes de toutes les variables des cliques et séparateurs auxquels ils sont associés. Dans la suite, nous noterons ϕ les potentiels des séparateurs et ψ ceux des cliques.

Le problème auquel nous allons nous intéresser est celui du calcul des probabilités marginales *a posteriori*, celui des probabilités *a priori* n'étant pas en soi extrêmement intéressant dans la mesure où il n'est effectué qu'une seule fois (pendant la phase d'initialisation). Considérons donc la portion d'arbre de jonction de la figure 6. Supposons qu'à l'issue de la phase d'initialisation, les potentiels de C_i, C_j et S sont respectivement $\psi(C_i) = P(C_i)$, $\psi(C_j) = P(C_j)$ et $\phi(S) = P(S)$. Puisque S est

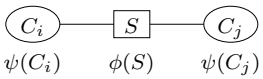


Figure 6. *L'algorithme d'inférence de Jensen*

un séparateur, $S = C_i \cap C_j$ et donc $\phi(S) = \sum_{C_i \setminus S} P(C_i) = \sum_{C_j \setminus S} P(C_j)$. On dit alors que l'arbre a atteint un état d'équilibre dans la mesure où tous les potentiels des cliques et des séparateurs sont cohérents entre eux.

Maintenant, on apprend une nouvelle information et celle-ci a pour effet de modifier $\psi(C_i)$ en $\psi^*(C_i)$. Ce peut être, par exemple, une information e_X indiquant qu'une variable X de C_i, qui pouvait *a priori* prendre les valeurs $x_1, \ldots, x_k$, ne peut plus prendre la valeur x_1, ou bien encore que l'on vient d'observer qu'une variable Y avait pris la valeur y. **Dans tous les cas, nous supposerons que toute information e_X relative à un nœud X est indépendante conditionnellement à X des autres variables du réseau et des informations relatives à ces variables**. Revenons à notre nouvelle information qui a modifié $\psi(C_i)$ en $\psi^*(C_i)$. L'équilibre est rompu puisque $\phi(S) \neq \sum_{C_i \setminus S} \psi^*(C_i)$. Pour rétablir ce dernier, il suffit d'appliquer la double opération suivante, que l'on appelle une *absorption de C_i par C_j* :

1) $\phi^*(S) = \sum_{C_i \setminus S} \psi^*(C_i)$ remplace le potentiel $\phi(S)$ associé à S ;

2) $\psi^*(C_j) = \psi(C_j) \times \dfrac{\phi^*(S)}{\phi(S)}$ remplace le potentiel $\psi(C_j)$ associé à C_j.

Il est alors facile de voir qu'à l'issue de ces deux étapes, l'arbre est à nouveau dans un état d'équilibre. L'algorithme d'inférence de Jensen consiste à utiliser de manière systématique la procédure d'absorption pour rétablir l'équilibre dans tout l'arbre de jonction. Pour assurer que la totalité du graphe sera traitée correctement, [JEN 96] montre qu'il suffit de choisir au hasard une clique C, la racine, et de lui appliquer successivement les deux fonctions *Collecte* et *Distribution*, ci-dessous. Une fois le graphe à nouveau dans un état d'équilibre, la probabilité marginale *a posteriori* de chaque variable X_i peut être déterminée simplement en choisissant une clique C (resp. un séparateur S) contenant la variable X_i et en calculant $\sum_{C \setminus \{X_i\}} \psi^*(C)$ (resp. $\sum_{S \setminus \{X_i\}} \phi^*(S)$).

Algorithme 2 (Collecte sur une clique C_i) *pour toutes les cliques C_j adjacentes à C_i excepté, si elle existe, la clique qui a appelé la* Collecte *de C_i, faire :*

(1) appeller Collecte *sur C_j,*

(2) effectuer l'absorption de C_j par C_i.

Algorithme 3 (Distribution sur une clique C_i) *pour toutes les cliques C_j adjacentes à C_i excepté, si elle existe, la clique qui a appelé la* Distribution *de C_i, faire :*

(1) effectuer l'absorption de C_i par C_j,

(2) appeller Distribution *sur* C_j.

Ce qu'il est important de remarquer pour notre unification des algorithmes de Pearl et de Jensen c'est que la taille des calculs lors de l'absorption correspond à la taille des cliques. De plus, si $\psi^*(C_i)$ correspond à une probabilité jointe des variables de la clique C_i et des nouvelles informations insérées dans le graphe, alors $\psi^*(C_j)$ est aussi une probabilité jointe.

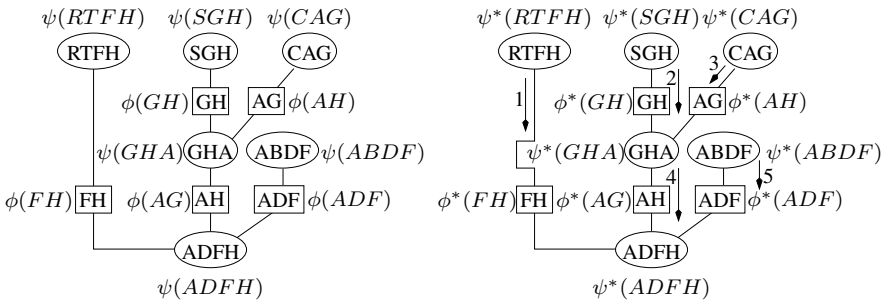


Figure 7. *Illustration de l'algorithme d'inférence de Jensen*

Illustrons cette méthode sur l'exemple de la figure 7 : Sur la partie gauche de la figure sont représentés les potentiels à l'issue de l'initialisation de l'arbre de jonction. De nouvelles informations e_T et e_C relatives à T et C sont apparues. Pour propager ces informations dans tout l'arbre de jonction, nous avons choisi arbitrairement la clique $ADFH$ comme racine. Nous appelons donc la fonction *Collecte* sur cette clique. Celle-ci appelle alors *Collecte* sur la clique $RTFH$. Cette clique n'ayant pas d'autre voisin, elle met à jour sa probabilité $\psi^*(RTFH) = P(R, T, F, H, e_T)$ et la clique $ADFH$ « absorbe » cette information : $\phi(FH)$ est remplacée par $\phi^*(FH) = P(F, H, e_T)$ et $\psi(ADFH)$ est remplacé par $\psi(ADFH) \times \phi^*(FH)/\phi(FH)$. La clique $ADFH$ exécute alors la fonction *Collecte* sur GHA qui, à son tour, l'exécute sur les cliques SGH puis CAG. SGH n'ayant reçu aucune information, $\psi^*(SGH) = \psi(SGH)$. La clique AGH absorbe alors les informations de SGH puis de CAG, elle a alors le potentiel $\psi^*(GHA)$, qu'elle transmet à $ADFH$ pour absorption. Enfin, cette dernière exécute *Collecte* sur $ABDF$ et absorbe ses informations de manière à obtenir le potentiel $\psi^*(ADFH)$. Dans une deuxième étape, on appelle la fonction *Distribution* sur $ADFH$ et l'équilibre est rétabli.

4. Une nouvelle triangulation pour graphe orienté

Dans la littérature sur les réseaux bayésiens, les triangulations sont exclusivement effectuées sur les graphes moralisés, c'est-à-dire sur des graphes non orientés. L'objectif de cet article étant de produire une variante de l'algorithme de Pearl effectuant des calculs similaires à ceux de Jensen, et la méthode de Pearl fonctionnant sur des graphes orientés, nous proposons dans cette section de réaliser une opération similaire

à la triangulation de la section précédente, mais reposant uniquement sur des graphes orientés. Pour cela, nous allons utiliser deux remarques à propos de cette triangulation.

Tout d'abord, nous avons vu qu'à chaque triangulation correspondait une séquence d'élimination de nœuds. Le corollaire du théorème 1 ci-dessous, dont on peut trouver la démonstration dans [LAU 96, proposition 2.17], montre que lorsque l'application de l'algorithme 1 avec une séquence d'élimination des nœuds particulière ne produit aucun lien supplémentaire, c'est que le graphe était déjà triangulé.

Corollaire 1 *Un graphe non orienté $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ est triangulé si et seulement si il existe une séquence d'élimination des nœuds σ telle que l'application de l'algorithme 1 avec cette séquence ne crée aucun lien n'appartenant pas déjà à $\mathcal{E}$.*

Par exemple, si l'on considère le graphe de la figure 8, l'application de la séquence d'élimination $\{S, C, G, R, T, B, D, H, F, A\}$ ne produit aucun lien, et, par conséquent, on peut en déduire que le graphe est triangulé.

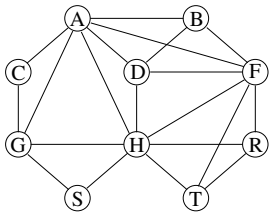
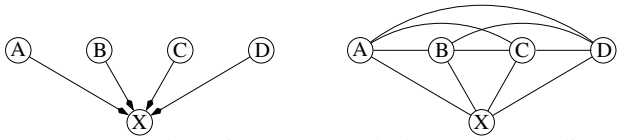


Figure 8. *Un graphe triangulé avec la séquence $\{S, C, G, R, T, B, D, H, F, A\}$*

La deuxième remarque qui va nous permettre d'effectuer la triangulation sur des graphes orientés concerne la phase de moralisation. On peut noter que la moralisation d'un graphe constitué uniquement d'un nœud et de ses parents forme une clique (cf. la figure 9).



(a) un réseau bayésien (b) le graphe moralisé

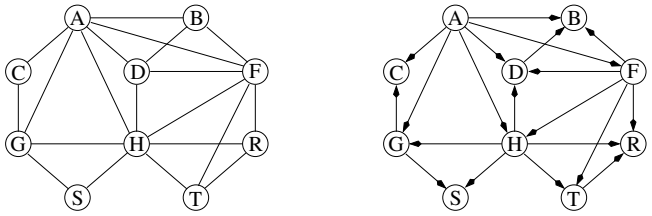
Figure 9. *La moralisation d'un graphe peut aboutir à une clique*

Reconsidérons maintenant l'algorithme de triangulation : lors de l'élimination du premier nœud, appelons-le X, on créait une clique contenant X et ses voisins. Donc, d'après le paragraphe précédent, si tous les voisins de X sont ses parents, la triangulation ne rajoute aucun lien (puisque la moralisation l'a déjà fait). Après l'élimination de X, on élimine Y. Là encore, si ses voisins sont ses parents, la moralisation a déjà créé

tous les liens nécessaires pour former une clique, la triangulation ne rajoutera donc aucun nouveau lien, etc. D'après la première remarque, cela implique que le graphe moralisé est déjà un graphe triangulé. Ces observations justifient la proposition suivante :

Proposition 1 *Soit $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ un graphe orienté, $\mathcal{V} = \{X_1, \ldots, X_n\}$. Soit σ une séquence d'élimination aboutissant à un graphe non orienté triangulé $\mathcal{G}_T = (\mathcal{V}, \mathcal{E})$. Alors la moralisation du graphe $\mathcal{G}' = (\mathcal{V}, \mathcal{A}')$, où $\mathcal{A}' = \{(X_i, X_j) : \sigma^{-1}(i) > \sigma^{-1}(j)$ et l'arête $(X_i, X_j) \in \mathcal{E}\}$, fournit un graphe déjà triangulé selon la séquence σ. On appellera $\mathcal{G}'$ un graphe orienté triangulé.*

Cette proposition indique simplement comment trouver un graphe triangulé orienté à partir d'un graphe triangulé non orienté : il suffit de partir de ce dernier, de parcourir les nœuds dans leur ordre d'élimination, et pour chaque lien adjacent au nœud, d'orienter ce lien vers le nœud lui-même. Par exemple, sur la figure 10a, considérons un graphe non orienté, triangulé selon la séquence $\{S, C, G, R, T, B, D, H, F, A\}$. Le nœud S est lié à G et H, on remplace donc les liens (G, S) et (H, S) par des arcs de G vers S et de H vers S. Ensuite, l'élimination de C provoque le remplacement des liens (A, C) et (G, C) par des arcs allant vers C. Plus intéressant, lorsqu'on passe à G, il ne reste plus que deux liens, à savoir (A, G) et (H, G), car (G, C) et (G, S) sont des arcs et non pas des liens. On remplace donc (A, G) et (H, G) par des arcs allant vers G, et ainsi de suite. Après avoir parcouru toute la séquence d'élimination, on obtient le graphe de la figure 10b. Le lecteur pourra vérifier aisément que ni la moralisation ni la triangulation ne rajouteront de lien à ce graphe. En ce sens, on peut dire que c'est un graphe orienté « triangulé ».



(a) graphe triangulé non orienté (b) graphe triangulé orienté

Figure 10. *Passage d'un graphe triangulé non orienté à un graphe triangulé orienté*

Bien entendu, le graphe $\mathcal{G}' = (\mathcal{V}, \mathcal{A}')$ ainsi obtenu représente une décomposition d'une loi de probabilité jointe $P'(\mathcal{V})$ différente de la loi $P(\mathcal{V})$ du réseau initial $\mathcal{G} = (\mathcal{V}, \mathcal{A})$. Le problème auquel nous sommes confrontés maintenant est de savoir si cette nouvelle décomposition est compatible avec celle du réseau d'origine, autrement dit P (donnée) est-elle aussi décomposable selon le nouveau graphe, ou encore existe-t-il une loi P' telle que $P(\mathcal{V}) = P'(\mathcal{V})$? Deux décompositions peuvent en effet être différentes sans que leurs lois de probabilité jointe le soient,

par exemple, lorsque A, B et C sont des variables indépendantes les unes des autres, $P(A,B,C) = P(A)P(B,C) = P(A,B)P(C)$. Cela dit, rien ne garantit *a priori* que $P'(\mathcal{V}) = P(\mathcal{V})$; il se pourrait en effet que deux variables dépendantes selon $\mathcal{G}$ soient indépendantes selon la décomposition de $\mathcal{G}'$. Dans ce cas, notre triangulation orientée n'offrirait que peu d'intérêt puisque des calculs effectués dans $\mathcal{G}'$, on ne pourrait rien déduire sur les probabilités des nœuds distribués selon la loi P. Fort heureusement, il y a compatibilité. En fait, la décomposition fournie par $\mathcal{G}'$ se contente simplement « d'occulter » quelques indépendances. Pour montrer cela, il suffit de constater que si l'on rajoute des arcs dans un réseau bayésien, la décomposition de la loi ainsi obtenue est compatible avec l'ancienne loi (on a simplement caché des indépendances), et de noter que l'on peut changer le sens d'un arc pourvu que l'on utilise les formules spécifiées dans le théorème suivant dû à [SHA 86] :

Théorème 2 (Renversement d'un arc (X, Y)) *Considérons un réseau bayésien $(\mathcal{V}, \mathcal{A}, \mathcal{P})$ dans lequel X est parent de Y (cf. figure 11a) et dans lequel il n'existe pas d'autre chemin de X vers Y (un chemin est un ensemble de nœuds $\{Z_1, \ldots, Z_k\}$ tel que $Z_1 = X$, $Z_k = Y$ et, pour tout $i \in \{1, \ldots, k-1\}$, $(Z_i, Z_{i+1}) \in \mathcal{A}$). Notons Pa les parents d'un nœud dans ce graphe. Selon la figure 11a, $Pa(X) = \{Z_i : i \in I\} \cup \{V_j : j \in J\}$ et $Pa(Y) = \{X\} \cup \{V_j : j \in J\} \cup \{W_k : k \in K\}$.*

Alors le réseau bayésien $(\mathcal{V}, \mathcal{A}', \mathcal{P}')$ obtenu en remplaçant l'arc (X, Y) par (Y, X) et en rajoutant les arcs (W_k, X), $k \in K$, et (Z_i, Y), $i \in I$, est encore un réseau bayésien, i.e. il ne contient pas de circuit. De plus, il vérifie les propriétés suivantes :

- $Pa'(Y) = (Pa(X) \cup Pa(Y)) \backslash \{X\}$;
- $Pa'(X) = (Pa(X) \cup Pa(Y) \cup \{Y\}) \backslash \{X\}$;
- $P'(Y|Pa'(Y)) = \sum_X P(Y|Pa(Y))P(X|Pa(X))$;
- $P'(X|Pa'(X)) = \dfrac{P(Y|Pa(Y)P(X|Pa(X))}{P'(Y|Pa'(Y))}$.

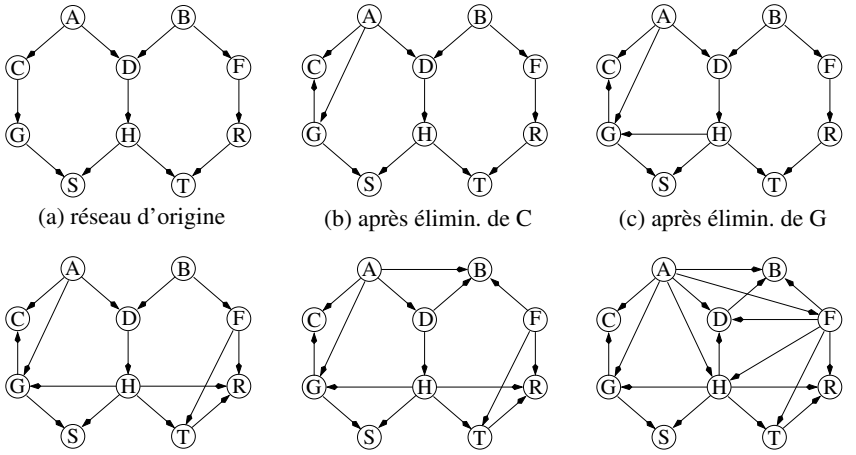
Enfin, $P'(\mathcal{V}) = P(\mathcal{V})$ pour toutes les valeurs des variables de $\mathcal{V}$, c'est-à-dire que les deux réseaux représentent la même loi de probabilité.

(a)graphe avant le renversement (b) graphe après le renversement

Figure 11. *Le renversement d'arc selon Shachter*

Revenons à notre problème : le réseau orienté triangulé représente-t-il une décomposition compatible avec celle du réseau de départ ? Eh bien oui car il suffit

de passer progressivement du graphe initial au réseau final en faisant des renversements d'arcs (en appliquant les formules ci-dessus) et, éventuellement, en rajoutant des arcs. Par exemple, essayons de transformer le réseau de la figure 12a en celui de la figure 12f. On avait vu que ce dernier était obtenu grâce à la séquence d'élimination $S, C, G, R, T, B, D, H, F, A$. Reprenons donc cette séquence et appliquons-lui des renversements d'arcs. Ceux adjacents à S sont bien orientés. Passons à C : l'arc (A, C) est bien orienté, mais pas (G, C). On renverse donc (G, C) (ce qui rajoute l'arc (A, G)) et l'on remarque que cela ne crée pas de circuit dans le graphe (cf. figure 12b). Passons à G : tous ses arcs sont bien orientés mais, pour obtenir le graphe final, il manque l'arc (H, G). On rajoute donc celui-ci et l'on met à jour la table de probabilité conditionnelle de G en dupliquant celle-ci autant de fois que H a de valeurs (cf. figure 12c). Passons à l'élimination du nœud R : l'arc (F, R) est bien orienté, mais pas (R, T). On applique donc le renversement d'arc sur (R, T), ce qui rajoute les arcs (F, T) et (H, R) (cf. figure 12d). L'élimination de T ne change pas le graphe. L'élimination de B revient à intervertir les arcs (B, F) et (B, D), ce qui induit l'ajout de l'arc (A, B) (cf. figure 12e) et ainsi de suite jusqu'à ce que l'on ait parcouru toute la séquence d'élimination.



(a) réseau d'origine (b) après élimin. de C (c) après élimin. de G

(d) après élimin. de R (e) après élimin. de B (f) graphe triangulé orienté

Figure 12. *Triangulation par renversement d'arc*

En fait, le processus ci-dessus s'applique à n'importe quel réseau bayésien et à n'importe quelle séquence d'élimination, ce qu'exprime la proposition suivante :

Proposition 2 *Soit $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ un DAG, $\mathcal{V} = \{X_1, \ldots, X_n\}$. Soit σ une séquence d'élimination aboutissant à un graphe triangulé non orienté $\mathcal{G}_E = (\mathcal{V}, \mathcal{E})$ et à un graphe triangulé orienté $\mathcal{G}_T = (\mathcal{V}, \mathcal{A}_T)$. Soit i_0 le plus grand index dans $\{1, \ldots, n-1\}$ tel que, pour tout $i < i_0$, l'ensemble des arcs de $\mathcal{A}$ adjacents à $X_{\sigma(i)}$ est égal à l'ensemble des arcs adjacents à $X_{\sigma(i)}$ dans $\mathcal{A}_T$. Si $\mathcal{V}' = \{Y : (X_{\sigma(i_0)}, Y) \in \mathcal{A} \backslash \mathcal{A}_T\}$*

est non vide, alors il existe un nœud $Z \in \mathcal{V}'$ sans ancêtre[6] *dans $\mathcal{V}'$ et le théorème 2 est applicable dans $\mathcal{G}$ sur l'arc $(X_{\sigma(i_0)}, Z)$. De plus, si l'on retriangule le graphe obtenu après application du théorème 2, on retrouve $\mathcal{G}_E$ et $\mathcal{G}_T$.*

Une application récursive de la proposition ci-dessus permet de déduire et de justifier l'algorithme ci-dessous :

Proposition 3 (graphe orienté triangulé) *Soient $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ un graphe orienté, $\mathcal{V} = \{X_1, \ldots, X_n\}$, et σ une séquence d'élimination aboutissant à un graphe orienté triangulé $\mathcal{G}_T = (\mathcal{V}, \mathcal{A}_T)$. L'application de la fonction suivante successivement à toutes les variables de $\mathcal{V}$ en suivant l'ordre défini par σ transforme $\mathcal{G}$ en $\mathcal{G}_T$:*

`fonction elimination`(*nœud* X)
1 `tant que` $\mathcal{V}' = \{Y : (X, Y) \in \mathcal{A} \backslash \mathcal{A}_T\}$ *est non vide* `faire`
2 *choisir un $Z \in \mathcal{V}'$ tel que $\nexists\, Y \in \mathcal{V}'$: Y est un ancêtre de Z*
3 *renverser dans $\mathcal{A}$ l'arc (X, Z) en appliquant le théorème 2 (cf. fig 11)*
4 *recalculer P en appliquant les formules de [SHA 86]*
5 `fait`
6 `pour tous` *les arcs (Y, X) de $\mathcal{A}_T \backslash \mathcal{A}$ adjacents à X* `faire`
7 *rajouter ces arcs à $\mathcal{A}$*
8 *mettre à jour la table de probabilité conditionnelle du nœud X en la dupliquant autant de fois que Y a de valeurs (i.e. rajouter une dimension en Y à la table de probabilité conditionnelle de X)*
9 `fait`

5. Une nouvelle variante de l'algorithme de Pearl

La proposition 3 permet de modifier un réseau bayésien en un réseau orienté triangulé $\mathcal{G}_T$ représentant la même loi de probabilité et, de plus, elle fournit les probabilités de chacun des nœuds/variables conditionnellement à leurs parents dans le graphe $\mathcal{G}_T$. Or c'est précisément ce dont ont besoin les algorithmes à la Pearl pour fonctionner. Il est donc légitime de se demander si leurs calculs seront plus efficaces dans le nouveau réseau que dans l'ancien. Comme nous allons le voir, non seulement ils le seront, mais ils vont atteindre la même complexité que ceux de Jensen. Pour ces calculs de complexité, nous ne tiendrons pas compte des phases d'initialisations distinctes nécessaires aux deux méthodes (car celles-ci ne sont effectuées qu'une seule fois et non à chaque propagation d'information).

Dans ce qui suit, nous n'allons pas utiliser le véritable algorithme de Pearl mais plutôt une variante, celle de [PEO 91], plus adaptée pour les cycles. À l'instar de l'algorithme de Jensen, les méthodes de calcul à la Pearl ne fournissent des résultats

6. une variable Y est ancêtre d'une autre variable Z s'il existe un ensemble de nœuds $\{A_1, \ldots, A_k\}$ tels que $A_1 = Y$, $A_k = Z$ et, pour tout $i \in \{1, \ldots, k-1\}$, il existe un arc de A_i vers A_{i+1}.

exacts que lorsque le graphe servant de support aux calculs ne contient aucun cycle. C'est pourquoi, pour étendre son algorithme à des réseaux quelconques, [PEA 88] proposa une modification de sa méthode d'origine appelée *coupe-cycle global*. Elle reposait sur l'idée que lorsqu'une variable est instanciée, on peut supprimer ses arcs sortants (c'est-à-dire les arcs vers les enfants) sans que cela rende la nouvelle décomposition incompatible avec la loi de probabilité d'origine. En instanciant suffisamment de variables, on peut donc supprimer tous les cycles du graphe (d'où le nom coupe-cycle) et utiliser l'algorithme de Pearl sur ce nouveau graphe. Enfin, il est bien connu que si une variable A peut prendre deux valeurs, a et $\overline{a}$, alors $P(B) = P(A = a, B) + P(A = \overline{a}, B)$. Par conséquent, pour obtenir les résultats sur le réseau avec cycles, il suffit de faire des sommes sur toutes les instanciations possibles des variables de coupe. L'inconvénient d'une telle méthode est qu'elle engendre assez rapidement une explosion combinatoire. En effet, si l'on coupe six variables pouvant prendre 10 valeurs, il y aura un million d'instanciations différentes et l'on devra donc effectuer puis sommer un million de calculs d'inférences. Même si ces derniers sont réalisés rapidement, globalement les temps de calculs peuvent être assez prohibitifs. [DIE 96] remarqua que certains de ces calculs étaient identiques quelles que soient les instanciations de certaines variables. Il en déduisit un algorithme d'inférence plus efficace, ne prenant en compte, localement, qu'un sous-ensemble des variables de coupe : le *coupe-cycle local*. [FAÕ0] a justifié mathématiquement sa validité. Dans ce qui suit nous allons utiliser une légère variante de [FAÕ0]. Précisément, la différence réside dans le stockage des calculs : [FAÕ0] utilise des ensembles de vecteurs, et nous allons plutôt utiliser des matrices, l'avantage étant que cela nous permettra de proposer des formules simples pour nos calculs, comme nous le verrons dans la proposition 4.

5.1. *Les calculs de [PEO 91] dans des graphes sans cycle*

Les méthodes à la Pearl, comme celle de [PEO 91], réalisent leurs calculs directement sur le réseau bayésien d'origine. En particulier, les nouvelles informations concernant les variables aléatoires sont insérées dans les nœuds correspondants. Par exemple, si l'on apprend l'information e_X = « X ne peut plus prendre la valeur x_1 », celle-ci est rajoutée comme indiquée sur la figure 13 et la probabilité conditionnelle de X est modifiée en $P(X, e_X|Pa(X))$. Il est utile de noter ici que, dans la méthode originelle de Pearl, la probabilité conditionnelle de X était transformée en $P(X|Pa(X), e_X)$. Bien évidemment, il est facile de passer de la formule de [PEO 91] à celle de Pearl : on sait que $\sum_X P(X|Pa(X), e_X) = 1$, par conséquent :

$$P(X|Pa(X), e_X) = \frac{P(X, e_X|Pa(X))}{\sum_X P(X, e_X|Pa(X))}.$$

Dans un graphe sans cycles, chaque arc sépare le graphe en deux sous-graphes disjoints. Si l'on note e l'ensemble des informations nouvelles sur les variables, pour chaque arc (X, Y), on peut partitionner e en deux sous-ensembles e^+_{XY} et e^-_{XY} selon

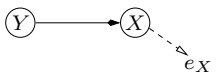


Figure 13. *L'insertion d'informations dans le réseau chez [PEO 91]*

que les informations sont insérées dans le sous-graphe du côté de X ou dans celui du côté de Y (cf. figure 14).

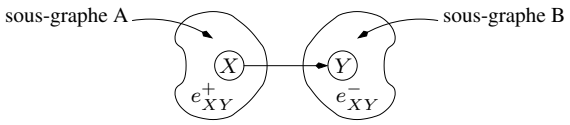


Figure 14. *Le partitionnement des informations dans le réseau*

[PEO 91] remarque alors que, pour que les informations e^+_{XY} soient transmises à Y, et donc incidemment à la partie B du graphe, il suffit que X envoie à Y (en descendant l'arc (X, Y)) un message $P(X, e^+_{XY})$. De même, Y peut transmettre à X un message $P(e^-_{XY}|X)$ et ainsi propager les informations de la partie B du graphe à la partie A. Des indépendances conditionnelles encodées dans le réseau assurent que ces messages sont indépendants et que leur produit est égal à $P(X, e^+_{XY}, e^-_{XY})$, ou encore à $P(X, e)$. L'algorithme de [PEO 91] consiste simplement à transmettre ces messages sur l'ensemble des arcs du réseau, et à effectuer le produit des messages transitant sur chaque arc. Pour bien séparer ces derniers, nous appliquerons la terminologie de Pearl, à savoir : le message descendant l'arc sera nommé π et l'autre λ. L'algorithme de [PEO 91] peut donc être décrit précisément de la manière suivante :

Algorithme 4 (Algorithme de Peot et Shachter pour des arbres) *Soit $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ un réseau bayésien ne contenant pas de cycles. Soit R un nœud quelconque de $\mathcal{V}$. On peut calculer les probabilités marginales* a posteriori *de chaque nœud en appliquant à R successivement les deux fonctions ci-dessous :*

Collecte sur un nœud X :

(1) pour tous les nœuds Y adjacents à X excepté, le cas échéant, le nœud qui a appelé Collecte *sur X, appeler* Collecte *sur Y,*

(2) si $X \neq R$, envoyer au nœud Y qui a appelé Collecte *sur X le message $\pi_Y(X)$ si X est parent de Y, et $\lambda_X(Y)$ sinon (voir ci-dessous les expressions de π et λ).*

Distribution sur un nœud X : *pour tous les nœuds Y adjacents à X excepté, le cas échéant, le nœud qui a appelé* Distribution *sur X, faire :*

(1) envoyer à Y un message $\pi_Y(X)$ si X est parent de Y, et $\lambda_X(Y)$ sinon,

(2) appeler Distribution *sur Y.*

Pour le graphe générique de la figure 15, les messages π et λ sont fabriqués de la manière suivante :

$$- \pi_Y(X) = \sum_{Pa(X)} \left[P(X, e_X | Pa(X)) \times \prod_{i=1}^{k} \pi_X(U_i) \times \prod_{j=1}^{m} \lambda_{Y_j}(X) \right] ;$$

$$- \lambda_Y(X) = \sum_{Y, Pa(Y)\setminus\{X\}} \left[P(Y, e_Y | Pa(Y)) \times \prod_{i=1}^{r} \pi_Y(V_i) \times \prod_{j=1}^{p} \lambda_{S_j}(Y) \right] ;$$

$$- \pi_Y(X) = P(X, e^+_{XY}), \quad \lambda_Y(X) = P(e^-_{XY} | X), \quad P(X, e) = \pi_Y(X)\lambda_Y(X).$$

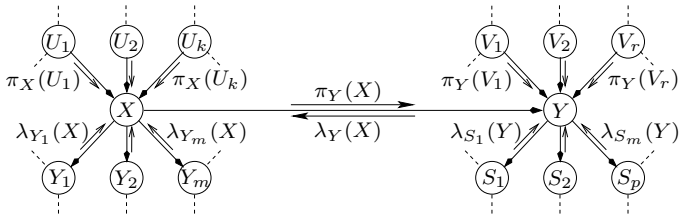


Figure 15. *Les calculs chez Peot et Shachter*

Le lecteur aura certainement remarqué la similitude entre l'algorithme ci-dessus et celui de Jensen : les deux méthodes utilisent deux phases, *Collecte* et *Distribution*. L'algorithme de Peot et Shachter est très simple à implémenter puisque, pour générer un message $\pi - \lambda$ envoyé par un nœud X à un nœud Y, il suffit de multiplier les messages provenant de tous les arcs adjacents à X, excepté l'arc entre X et Y, par la probabilité conditionnelle stockée en X. Malheureusement, pour fonctionner correctement, cet algorithme requiert que les messages envoyés à un nœud soient tous indépendants les uns des autres, ce qui n'est vrai que lorsque le graphe ne contient pas de cycle (voir [SUE 90] pour une explication détaillée du phénomène). C'est pourquoi une adaptation telle qu'un coupe-cycle est nécessaire.

5.2. *Le coupe-cycle local et l'algorithme de Peot et Shachter*

Le coupe-cycle consiste à effectuer les calculs de Peot et Shachter, mais en conservant les dimensions des variables de coupe. Autrement dit, on n'effectue jamais les sommations sur les variables de coupe décrites dans les formules en haut de cette page et en bas de la page précédente. En conséquence, il est important de bien choisir ces dernières si l'on veut éviter une explosion combinatoire des calculs. Divers algorithmes ont été développés à cet effet ([BAR 94], [BEC 96a], [BEC 99a]). Nous verrons dans la sous-section suivante que, dans nos réseaux bayésiens triangulés, un algorithme de sélection des variables de coupe émerge tout naturellement.

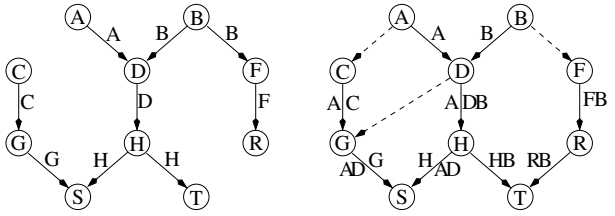
Afin de mieux comprendre quelles sont effectivement les sommations réalisées dans les calculs, nous allons « labelliser » les arcs et indiquer dans ces labels les

dimensions et la nature des messages transitant sur ces arcs. Remarquons tout d'abord que, pour des graphes sans cycles, les messages $\pi_Y(X)$ et $\lambda_Y(X)$ n'ont qu'une seule dimension, X, puisqu'ils correspondent respectivement à $P(X, e^+_{XY})$ et $P(e^-_{XY}|X)$. On obtient alors des labels similaires à ceux notés à côté des arcs sur la figure 16a. Autrement dit, si l'on appelle C_{XY} le label de l'arc (X, Y), les formules de Peot et Shachter peuvent être énoncées de la manière suivante :

Les messages $\pi - \lambda$ avec labels :

$$- \pi_Y(X) = \sum_{\mathcal{V}\setminus C_{XY}} \left[P(X, e_X|Pa(X)) \times \prod_{i=1}^{k} \pi_X(U_i) \times \prod_{j=1}^{m} \lambda_{Y_j}(X) \right] ;$$

$$- \lambda_Y(X) = \sum_{\mathcal{V}\setminus C_{XY}} \left[P(Y, e_Y|Pa(Y)) \times \prod_{i=1}^{r} \pi_Y(V_i) \times \prod_{j=1}^{p} \lambda_{S_j}(Y) \right].$$



(a) un graphe sans cycle (b) un graphe avec cycles

Figure 16. *La labellisation des arcs*

Lorsque le graphe contient des cycles, on choisit des nœuds de coupe que l'on instancie. Sur la figure 16b, nous avons choisi arbitrairement les nœuds A, B et D. Il n'est pas nécessaire de couper l'ensemble des arcs sortant de ces nœuds : il suffit juste d'en couper suffisamment pour que le graphe devienne acyclique tout en conservant sa connexité. Nous avons donc choisi de couper les arcs (A, C), (B, F) et (D, G), ce que nous avons représenté par des arcs en pointillés. [FAõ0] montre que la labellisation ci-dessous permet à l'algorithme de Peot et Shachter (avec les messages avec labels) d'effectuer des calculs de probabilité exacts (les messages ne sont bien évidemment envoyés que sur les arcs non coupés) :

Algorithme 5 (labellisation) *Soit $(\mathcal{V}, \mathcal{A}, \mathcal{P})$ un réseau bayésien. On affecte à chaque arc (X, Y) de $\mathcal{A}$ le label X, puis tant qu'il reste des cycles, on en choisit un, on coupe un de ses arcs, appelons-le (A, B), et on rajoute à chaque arc du cycle le label de l'arc (A, B).*

Ainsi, sur le graphe de la figure 16b, on a coupé le cycle $ADHSGC$ avec l'arc (A, C). On a donc tout naturellement rajouté le label A aux arcs (A, D), (D, H), (H, S), (G, S) et (C, G). De même, on a coupé le cycle $BFRTHD$ avec l'arc (B, F), ce qui a entraîné le rajout du label B aux arcs (F, R), (R, T), (H, T), (D, H) et

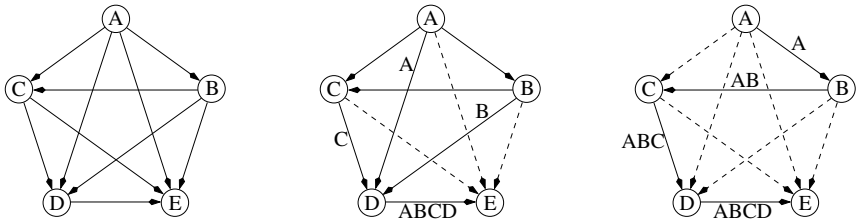
(B, D). Enfin, on a coupé le cycle $DHSG$ avec l'arc (D, G), ce qui a provoqué le rajout du label D aux arcs (D, H), (H, S) et (G, S). Finalement, un arc comme (D, H) aura le label A, B, D. Si l'on applique l'algorithme de Peot et Shachter (avec les messages avec labels), les messages π et λ de l'arc (D, H) seront donc des matrices de dimension 3 (dimensions relatives à A, D et B).

Algorithme 6 (de Faÿ et Jaffray) *Soit $(\mathcal{V}, \mathcal{A}, \mathcal{P})$ un réseau bayésien. Pour calculer les probabilités marginales* a posteriori *de toutes les variables, on labellise les arcs grâce à l'algorithme 5. Ensuite on choisit arbitrairement un nœud R et on applique les fonctions* Collecte *puis* Distribution *de l'algorithme 4 avec les messages $\pi - \lambda$ avec labels.*

5.3. *Une nouvelle variante de l'algorithme de Pearl*

Nous avons maintenant tout le matériel théorique à notre disposition pour développer une méthode de Pearl pouvant rivaliser avec celle de Jensen : il suffit simplement d'appliquer l'algorithme 6 dans les réseaux bayésiens triangulés de la section 4. Il faut tout de même préciser comment effectuer la coupe puisque celle-ci n'est pas unique. Pour cela, rappelons-nous que, dans le graphe orienté triangulé, chaque variable X_i et l'ensemble des nœuds Y_j liés à X_i par des arcs orientés vers X_i forment une clique. Exploitons ce fait pour effectuer la coupe, et considérons donc la clique de la figure 17a. Puisque le graphe est un DAG, il existe un nœud, ici E, qui n'a que des parents dans la clique. Si l'on élimine ce nœud ainsi que ses arcs adjacents, il reste encore une clique, $ABCD$. Il existe donc à nouveau un nœud n'ayant que des parents, D. En itérant, on peut classer les nœuds dans l'ordre E, D, C, B, A. Comme nous sommes dans une clique, toutes les variables sont reliées entre elles par des arcs. Parmi l'ensemble des cycles passant par E, on peut donc sélectionner des cycles formés par trois nœuds. Parmi ceux-ci, nous ne considérerons que les cycles contenant les nœuds D et E, autrement dit les cycles ADE, BDE et CDE. Si l'on coupe ceux-ci respectivement sur les arcs (A, E), (B, E) et (C, E), c'est-à-dire sur les arcs provenant de tous les parents excepté le premier à être éliminé après E, on obtient les labels de la figure 17b. Il est remarquable que seul le label de l'arc (D, E) a été modifié par ces coupes et, de plus, il est inclus dans l'ensemble des variables de la clique. Remarquons en outre qu'il n'existe plus de cycle passant par E et, *a fortiori*, par l'arc (D, E). On peut réitérer ce procédé sur D : on considère tous les cycles de longueur 3 passant par D et C, à savoir ACD et BCD et on coupe les arcs (A, D) et (B, D), etc. On obtient finalement les labels de la figure 17c.

Dans notre réseau bayésien triangulé, il suffit d'appliquer cette méthode sur l'ensemble des nœuds ordonnés suivant la séquence d'élimination. Cela garantit que les labels seront toujours inclus dans les cliques correspondantes de l'arbre de jonction et donc que l'algorithme de Faÿ et Jaffray effectué avec ces labels aura une complexité similaire à celle de Jensen. Ceci motive la proposition suivante :



(a) une clique (b) E n'est plus dans des cycles (c) coupes de la clique

Figure 17. *Les coupes effectuées dans les cliques*

Proposition 4 (variante de l'algorithme de Pearl) *Soient un réseau bayésien* $(\mathcal{V}, \mathcal{A}, \mathcal{P})$, $\mathcal{V} = \{X_1, \ldots, X_n\}$, *et* σ *une séquence d'élimination. Soit* $\mathcal{G}' = (\mathcal{V}, \mathcal{A}')$ *le réseau orienté triangulé résultant de la proposition 3. L'application de l'algorithme de Faÿ et Jaffray avec l'algorithme de coupe suivant fournit un calcul exact. De plus, la complexité des calculs est égale à celle obtenue par Jensen (avec la même séquence d'élimination* σ*).*

Algorithme de coupe : *Pour* i *variant de* 1 *à* n*, soit* j *le plus petit index tel que l'arc* $(X_{\sigma(j)}, X_{\sigma(i)})$ *appartient à* $\mathcal{A}'$*. Dans tous les cycles (triangles)* $X_{\sigma(i)}X_{\sigma(j)}X_{\sigma(k)}$*, couper l'arc* $(X_{\sigma(k)}, X_{\sigma(i)})$*.*

Enfin, les messages $\pi - \lambda$ *utilisés sont égaux à :*

$$\pi_Y(X) = P(C_{XY}, e^+_{XY}) \quad \textit{et} \quad \lambda_Y(X) = P(e^-_{XY}|C_{XY}).$$

En fait, les messages de cet algorithme s'apparentent à ceux de [SHE 90] dans la mesure où l'on n'effectue jamais de divisions, seulement des multiplications et des sommes. En ce sens, notre algorithme est légèrement plus rapide que celui de Jensen. En contrepartie, l'algorithme de Jensen ne stocke qu'une seule probabilité dans ses séparateurs et nous stockons deux messages dans nos arcs ; notre algorithme consomme donc plus de mémoire que celui de Jensen.

6. Conclusion et perspectives

Il est communément admis dans la littérature que les méthodes d'inférence non orientées sont supérieures en termes de rapidité de calculs aux méthodes orientées. Nous avons montré dans cet article que cela était dû au fait que ces dernières n'utilisaient pas dans leurs calculs de structures secondaires adaptées et que, dès lors qu'elles sont dotées d'une structure adaptée, elles deviennent aussi efficaces que les méthodes non orientées comme celle de Jensen.

D'un point de vue pratique, notre algorithme offre généralement des performances similaires à celles de Jensen mais il est des situations où il le surpasse. Remarquons en effet que, chez Jensen, les temps de calcul dépendent principalement des calculs

effectués dans les cliques. Or dans notre algorithme, certaines tables de probabilité conditionnelle ont précisément la taille des cliques de Jensen. Si l'on effectue les phases de collecte et de diffusion en passant par tous les nœuds du réseau comme chez Jensen, on ne peut raisonnablement espérer obtenir un algorithme plus rapide que ce dernier. Toutefois, une analyse en d-séparation telle que celle préconisée dans [GON 03] peut réduire de manière drastique la phase de collecte et ainsi rendre notre algorithme plus performant que celui de Jensen. Prenons par exemple l'arbre de jonction et la structure secondaire orientée de la figure 18, et supposons que l'on veuille propager deux informations e_A et e_F portant respectivement sur les variables A et F. L'analyse en d-séparation de [GON 03] montre que le réseau bayésien peut être scindé en trois zones (en pointillés) et que seuls des messages de A vers C, de C vers E, et de E et F vers G ont besoin d'être propagés pour calculer les probabilités marginales *a posteriori* (conditionnellement à e_A et e_F). Il n'y a donc pas à réaliser de phase de collecte. En revanche, Jensen a besoin de faire à la fois une phase de collecte et une phase de diffusion, d'où un temps de calcul plus élevé.

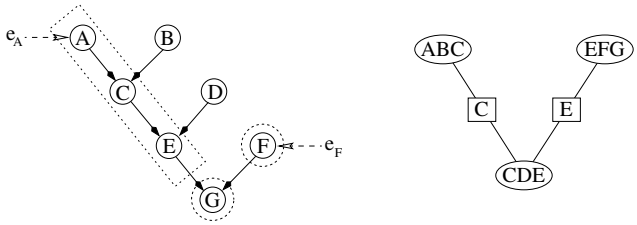
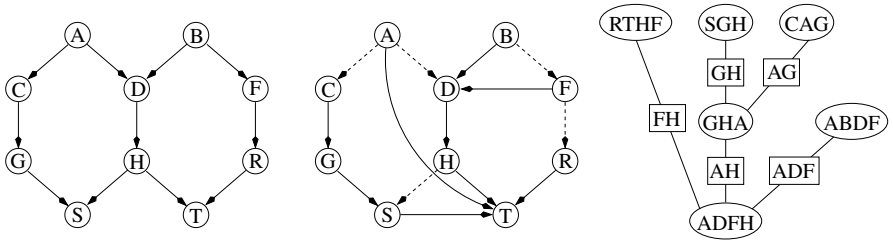


Figure 18. *Un arbre de jonction et un graphe triangulé orienté associé*



(a) réseau d'origine (b) une structure secondaire (c) l'arbre de jonction

Figure 19. *De nouvelles structures secondaires plus performantes ?*

Enfin, si notre structure secondaire orientée ne permet pas toujours de surpasser l'algorithme de Jensen, c'est principalement dû au fait que beaucoup d'arcs du réseau d'origine sont renversés, occultant par là même beaucoup d'indépendances probabilistes. On peut envisager pour l'avenir différentes structures secondaires qui minimiseraient les renversements d'arcs et pourraient donc tirer un meilleur parti des analyses en d-séparation. À titre d'exemple, si l'on rajoute au réseau bayésien de la figure 19a

(que nous avions déjà étudié dans les sections 3 et 4), les arcs (A, T), (F, D) et (S, T), il est bien évident que le nouveau réseau ainsi constitué sera compatible avec la loi de probabilité représentée par le réseau de la figure 19a puisque l'ajout des arcs a simplement occulté quelques indépendances. Or si, dans ce nouveau réseau, on effectue les coupe-cycles locaux symbolisés par les arcs en pointillés, on peut montrer que les calculs (collecte + diffusion sur l'ensemble des nœuds) d'une variante de l'algorithme de Pearl dans laquelle on génère les messages issus d'un nœud dans un certain ordre seront similaires (en termes de taille des calculs) à ceux de Jensen (dans l'arbre de jonction indiqué dans la figure 19c). Or la nouvelle structure orientée ainsi proposée n'ayant effectué aucun renversement d'arc, elle peut profiter au maximum des propriétés de d-séparation du réseau d'origine.

Remerciements

Nous tenons à exprimer toute notre gratitude aux professeurs Jean-Yves Jaffray et Khaled Mellouli pour tout le temps qu'ils nous ont consacré en discussions et en conseils concernant cet article.

7. Appendice : démonstrations

Démonstration de la proposition 1 : Pour tout lien $(X_i, X_j) \in \mathcal{E}$, soit $\sigma^{-1}(i) < \sigma^{-1}(j)$ soit $\sigma^{-1}(j) < \sigma^{-1}(i)$. Dans le premier cas, on aura $(X_j, X_i) \in \mathcal{A}'$ et dans le second $(X_i, X_j) \in \mathcal{A}'$. Dans les deux cas, la moralisation de $\mathcal{G}' = (\mathcal{V}, \mathcal{A}')$ donne un graphe $\mathcal{G}'' = (\mathcal{V}, \mathcal{E}'')$ tel que le lien $(X_i, X_j) \in \mathcal{E}''$. Donc $\mathcal{E} \subseteq \mathcal{E}''$.

Moralisons $\mathcal{G}'$: commençons par relier les parents de $X_{\sigma(1)}$. D'après la définition de $\mathcal{A}'$, pour deux parents distincts $X_{\sigma(i)}$ et $X_{\sigma(j)}$, il existe dans $\mathcal{E}$ des liens entre $X_{\sigma(1)}$ et $X_{\sigma(i)}$ d'une part, et entre $X_{\sigma(1)}$ et $X_{\sigma(j)}$ d'autre part. Or le graphe $\mathcal{G}_T$ est triangulé et $X_{\sigma(1)}$ est le premier nœud éliminé. Donc $X_{\sigma(1)}$, $X_{\sigma(i)}$ et $X_{\sigma(j)}$ appartiennent à la même clique, et il existe donc aussi dans $\mathcal{E}$ un lien entre $X_{\sigma(i)}$ et $X_{\sigma(j)}$. Aussi le mariage des parents de $X_{\sigma(1)}$ ne peut rajouter de lien n'appartenant pas déjà à $\mathcal{E}$. Remarquons que $X_{\sigma(1)}$ ne peut, par définition, être parent d'un autre $X_{\sigma(i)}$ dans $\mathcal{A}'$.

Marions maintenant les parents de $X_{\sigma(2)}$. Puisque $X_{\sigma(1)}$ n'est pas parent de $X_{\sigma(2)}$, on peut réitérer l'argument du paragraphe précédent et déduire que le mariage des parents de $X_{\sigma(2)}$ n'ajoute aucun lien n'appartenant pas déjà à $\mathcal{E}$. Par récurrence sur les $X_{\sigma(i)}$, on déduit que $\mathcal{E}'' \subseteq \mathcal{E}$, et donc que $\mathcal{E}'' = \mathcal{E}$. Puisque $\mathcal{G}_T = (\mathcal{V}, \mathcal{E})$ est triangulé selon σ, le graphe moralisé de $\mathcal{G}'$ l'est aussi. ♦

Démonstration de la proposition 2 : Supposons que $\mathcal{V}' \neq \emptyset$ et qu'il existe un nœud $X_{\sigma(j)}$ dans $\mathcal{V}'$ tel que l'on ne puisse pas renverser l'arc $(X_{\sigma(i_0)}, X_{\sigma(j)})$. D'après le théorème 2, cela implique qu'il existe un chemin (autre que l'arc $(X_{\sigma(i_0)}, X_{\sigma(j)})$) $\{Z_1 = X_{\sigma(i_0)}, Z_2, \ldots, Z_p = X_{\sigma(j)}\}$ allant de $X_{\sigma(i_0)}$ à $X_{\sigma(j)}$. On sait que $j > i$. En effet, si ce n'était pas le cas, l'arc $(X_{\sigma(i_0)}, X_{\sigma(j)})$ appartiendrait à $\mathcal{A}_T$ puisque le nœud

$X_{\sigma(j)}$ serait éliminé avant $X_{\sigma(i_0)}$ (cf. la proposition 1). De même, l'arc (Z_{p-1}, Z_p) implique que Z_{p-1} est éliminé après $X_{\sigma(i_0)}$ car, dans le cas contraire, il correspondrait à un $X_{\sigma(k)}$, $k < i_0$, dont l'arc (Z_{p-1}, Z_p) ne pourrait appartenir à $\mathcal{A}_T$. Par récurrence, on déduit que l'ensemble des Z_i, $i > 1$, est éliminé après $X_{\sigma(i_0)}$.

Puisque $\mathcal{G}$ est un graphe sans circuit, il existe un nœud Z dans $\mathcal{V}'$ qui n'a pas d'ancêtre dans $\mathcal{V}'$. S'il existait un chemin autre que l'arc $(X_{\sigma(i_0)}, Z)$ allant de $X_{\sigma(i_0)}$ à Z, le deuxième nœud de ce chemin, Z_2, appartiendrait aussi à $\mathcal{V}'$ puisque, d'après le paragraphe précédent, ce nœud serait éliminé après $X_{\sigma(i_0)}$ et l'arc $(X_{\sigma(i_0)}, Z_2)$ appartiendrait à $\mathcal{A}\backslash\mathcal{A}_T$. D'où une contradiction puisque Z_2 serait ancêtre de Z. On peut donc appliquer le théorème 2 et renverser le sens de l'arc $(X_{\sigma(i_0)}, Z)$.

(a) graphe avant le renversement (b) graphe après le renversement

Figure 20. *Les arcs rajoutés par le renversement de l'arc* (X, Y)

Montrons maintenant que les arcs rajoutés par le renversement de l'arc $(X_{\sigma(i_0)}, Z)$ correspondent à des liens dans $\mathcal{E}$. La figure 20 montre quels sont ces arcs : il s'agit de relier tous les parents de $X_{\sigma(i_0)}$ à Z, et tous les parents de Z à $X_{\sigma(i_0)}$. Puisque, avant le renversement, $X_{\sigma(i_0)}$ et les W_k sont des parents de Z, pour obtenir $\mathcal{G}_E$, la phase de moralisation les relie. Par conséquent, il existe des liens entre ces nœuds dans $\mathcal{E}$. Donc l'ajout des arcs $(W_k, X_{\sigma(i_0)})$ correspond à des liens de $\mathcal{E}$. On sait que Z, les U_i, V_j et W_k sont éliminés après $X_{\sigma(i_0)}$ (voir l'argument du premier paragraphe de la démonstration). Or, lors de la triangulation (proposition 1), lorsque l'on élimine $X_{\sigma(i_0)}$, on relie entre eux tous les voisins de $X_{\sigma(i_0)}$ de manière à former une clique. Par conséquent, dans $\mathcal{E}$, il doit y avoir des liens entre les U_i et Z. Donc l'ensemble des arcs rajoutés par le renversement de $(X_{\sigma(i_0)}, Z)$ correspondent à des liens de $\mathcal{E}$. Maintenant, $X_{\sigma(i_0)}$ et Z ont de nouveaux parents. Pour s'assurer que, si on retriangule, on obtienne à nouveau $\mathcal{G}_E$, il suffit de montrer que le mariage des parents de $X_{\sigma(i_0)}$, et de Z, ne crée pas de lien n'appartenant pas à $\mathcal{E}$. Comme les U_i et les V_j (resp. les V_j et les W_k) étaient déjà des parents de $X_{\sigma(i_0)}$ (resp. de Z), la moralisation du graphe avant renversement les avait déjà mariés. On a vu qu'avant le renversement, $X_{\sigma(i_0)}$ et W_k étaient parents de Z. Donc, par moralisation, ils ont été reliés entre eux. Or, lors de l'élimination de $X_{\sigma(i_0)}$, on relie tous ses voisins. Comme les U_i sont des parents de $X_{\sigma(i_0)}$, ils sont aussi voisins. Donc l'élimination de $X_{\sigma(i_0)}$ crée nécessairement des liens dans $\mathcal{E}$ entre les U_i et les W_k. Par conséquent, une retriangulation du graphe obtenu à partir de $\mathcal{G}$ après renversement a pour résultat $\mathcal{G}_E$ et $\mathcal{G}_T$. ♦

Démonstration de la proposition 3 : Lorsque l'on appelle la fonction `elimination (nœud)` sur le nœud $X_{\sigma(1)}$, si $\mathcal{V}' = \emptyset$, tous les arcs adjacents à $X_{\sigma(1)}$ sont bien orientés, on ne pratique donc pas de renversement d'arc. En revanche, si $\mathcal{V}' = \{Y_1, \ldots, Y_p\}$ $\neq \emptyset$, la ligne 2 de la fonction permet de choisir un Y_i et d'appliquer la proposition 2.

Une triangulation du graphe ainsi obtenu résultera donc encore en $\mathcal{G}_T$. Notons que les arcs adjacents à $X_{\sigma(1)}$ et rajoutés à $\mathcal{A}$ sont orientés vers $X_{\sigma(1)}$. Ils n'appartiendront donc pas à $\mathcal{V}'$. Par conséquent, la boucle des lignes 1–5 élimine progressivement tous les arcs « mal » orientés et, à la fin de cette boucle, $\mathcal{V}' = \emptyset$. De plus, la proposition 2 assure que le graphe triangulé du graphe $\mathcal{G}$ ainsi modifié sera égal à $\mathcal{G}_T$.

Lorsque l'on effectue la boucle des lignes 6–9, on ne fait que rajouter des arcs $(Y, X_{\sigma(1)})$ orientés vers $X_{\sigma(1)}$. Commes ces arcs appartenaient déjà à $\mathcal{A}_T$, cela implique qu'ils correspondaient à des liens adjacents à $X_{\sigma(1)}$ dans le graphe non orienté triangulé ; appelons-le $\mathcal{G}_E$. Par conséquent, lors de l'élimination de $X_{\sigma(1)}$, tous les nœuds Y correspondants sont reliés aux autres voisins de $X_{\sigma(1)}$. Or l'ajout de ces arcs par les lignes 6–9 produit précisément cet effet par moralisation. Donc la triangulation après leur ajout résulte à nouveau en $\mathcal{G}_E$, et donc aussi en $\mathcal{G}_T$.

On peut reprendre les deux paragraphes ci-dessus pour $X_{\sigma(2)}$, $X_{\sigma(3)}$, etc. Par récurrence, on déduit qu'une fois les opérations des lignes 1–9 effectuées, on retrouve toujours le graphe triangulé $\mathcal{G}_T$. Or, les lignes 1–5 assurent que tous les arcs adjacents à chacun des nœuds du graphe $\mathcal{G}$ modifié sont orientés comme ceux de $\mathcal{G}_T$ et les lignes 6–9 assurent que tous les arcs de $\mathcal{G}_T$ appartiennent au graphe $\mathcal{G}$ modifié. On en déduit donc que $\mathcal{G}$ est transformé en $\mathcal{G}_T$. ♦

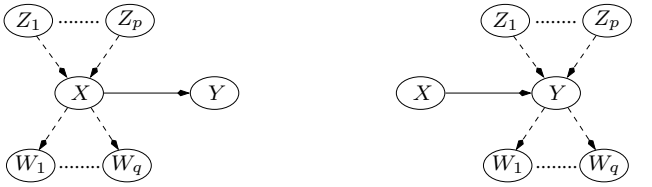
Démonstration de la proposition 4 : D'après la proposition 3, le graphe $\mathcal{G}'$ représente une décomposition compatible avec la loi jointe définie par $\mathcal{P}$. Par conséquent, si l'algorithme de coupe supprime tous les cycles, l'algorithme de Faÿ et Jaffray fournira un calcul exact.

D'après la proposition 1, tous les arcs de $\mathcal{A}'$ adjacents à $X_{\sigma(1)}$ sont orientés vers $X_{\sigma(1)}$. Notons les $(X_{\sigma(i_1)}, X_{\sigma(1)}), \ldots, (X_{\sigma(i_k)}, X_{\sigma(1)})$. Puisque k est fini, il existe parmi $\{i_1, \ldots, i_k\}$ un plus petit élément. Sans perte de généralité, supposons que ce soit i_1. Par triangulation, on sait que tous les voisins de $X_{\sigma(1)}$ sont liés entre eux pour former une clique. Donc, dans $\mathcal{A}'$, il existe un arc entre chaque couple de variables $X_{\sigma(i_1)}, X_{\sigma(i_j)}$, $j \in \{2, \ldots, k\}$. Par conséquent, il existe $k-1$ triangles $X_{\sigma(i_1)}X_{\sigma(i_j)}X_{\sigma(1)}$, et la coupe des arcs $(X_{\sigma(i_j)}, X_{\sigma(1)})$ supprime l'ensemble des cycles passant par $X_{\sigma(1)}$ (puisqu'il ne reste plus qu'un seul arc adjacent à $X_{\sigma(1)}$: $(X_{\sigma(i_1)}, X_{\sigma(1)})$). Passons maintenant à $X_{\sigma(2)}$. On sait qu'il n'existe pas d'arc $(X_{\sigma(1)}, X_{\sigma(2)})$ dans $\mathcal{A}'$, simplement parce que $X_{\sigma(1)}$ n'a que des parents. On peut donc recommencer avec $X_{\sigma(2)}$ le procédé appliqué à $X_{\sigma(1)}$. Par récurrence sur l'ensemble des nœuds ordonnés selon la séquence σ, on supprime tous les cycles du graphe. Donc l'algorithme de Faÿ et Jaffray fournit des calculs exacts.

Notons que, si on labellise chaque arc (X, Y) par X, l'algorithme de coupe dans l'énoncé de la proposition ne modifie qu'un seul label : celui du seul arc adjacent à $X_{\sigma(i)}$ non coupé et celui-ci devient égal à l'ensemble des parents de $X_{\sigma(i)}$. En effet, un triangle $X_{\sigma(i)}X_{\sigma(j)}X_{\sigma(k)}$ n'est coupé sur l'arc $(X_{\sigma(i)}, X_{\sigma(k)})$ que si $j < i$. Par conséquent, $X_{\sigma(j)}$ est éliminé avant $X_{\sigma(i)}$ et il existe donc un arc $(X_{\sigma(i)}, X_{\sigma(j)})$. Autrement dit, les arcs du triangle sont $(X_{\sigma(i)}, X_{\sigma(j)})$, $(X_{\sigma(i)}, X_{\sigma(k)})$ et $(X_{\sigma(j)}, X_{\sigma(k)})$. Donc la coupe de l'arc $(X_{\sigma(i)}, X_{\sigma(k)})$ ne rajoute bien que le label $X_{\sigma(i)}$ à l'arc $(X_{\sigma(j)}, X_{\sigma(k)})$.

Or, lorsqu'on a coupé tous les triangles passant par un nœud, il ne reste plus aucun cycle passant par lui, et *a fortiori* par son seul arc adjacent. Par conséquent, les coupes ultérieures ne modifieront plus le label de cet arc.

Montrons maintenant par récurrence que les messages π et λ de Faÿ et Jaffray correspondent à ceux mentionnés dans l'énoncé de la proposition. Base de la récurrence : supposons qu'un nœud X n'ait, après coupe, qu'un seul enfant, Y (cf. figure 21a). X ne peut avoir de parents $Z_1, \ldots, Z_p$ dans $\mathcal{G}'$ car l'algorithme de coupe aurait conservé un arc d'un parent vers X. Si X a reçu une information e_X, alors $e^+_{XY} = e_X$, sinon $e^+_{XY} = e_X = \emptyset$. D'après les formules $\pi - \lambda$ avec messages labellisés, le message $\pi_Y(X)$ est égal à $P(X, e_X)$ (il n'y a aucun produit à effectuer puisque tous les arcs adjacents à X ont été coupés, excepté l'arc (X, Y)). Or $C_{XY} = X$. En effet, si ce n'était pas le cas, il existerait un nœud $Z \neq X$ appartenant au label C_{XY}. D'après l'algorithme de coupe, cela signifierait qu'il existerait un triangle ZXY, où Z est parent de X et de Y, ce qui est impossible puisque X n'a pas de parent dans $\mathcal{G}'$. Donc $\pi_Y(X) = P(C_{XY}, e^+_{XY})$.



(a) X envoie un message (b) Y envoie un message

Figure 21. *Les bases de la récurrence*

Considérons maintenant un nœud Y n'ayant qu'un seul voisin après coupe : un parent (cf. figure 21b). D'après l'algorithme de coupe, Y ne peut avoir de fils $W_1, \ldots, W_q$ dans $\mathcal{G}'$ (car la coupe conserve toujours un des arcs). Puisque l'arc (X, Y) existe encore et que les arcs (Z_i, Y) ont été supprimés, d'après l'algorithme de coupe, X est nécessairement le premier parent de Y à être éliminé après lui lors de la triangulation. Par conséquent, d'après le troisième paragraphe de la démonstration, le label de l'arc (X, Y) est égal à l'ensemble des parents de Y, soit $C_{XY} = \{X, Z_1, \ldots, Z_p\}$. D'après Faÿ et Jaffray, le message envoyé est $\sum_{\mathcal{V} \setminus C_{XY}} P(Y, e^-_{XY} | Pa(Y))$ (il n'y pas de produit à effectuer puisque seul l'arc (X, Y) n'a pas été coupé). Par conséquent,

$$\lambda_Y(X) = \sum_Y P(Y, e^-_{XY} | X, Z_1, \ldots, Z_p) = P(e^-_{XY} | X, Z_1, \ldots, Z_p).$$

Autrement dit, $\lambda_Y(X) = P(e^-_{XY} | C_{XY})$.

Passons maintenant au cas général décrit sur la figure 22. Puisque les arcs (Z_i, X) sont coupés, les nœuds Z_i sont nécessairement éliminés après X lors de la triangulation. Pour la même raison, puisque X est parent de V_i, tous les arcs provenant des

autres parents de V_i dans $\mathcal{G}'$ ont été coupés dans des triangles faisant intervenir X. Or ce n'est possible que si les V_i sont éliminés avant X.

On sait par hypothèse de récurrence que $\lambda_X(V_i) = P(e^-_{XV_i}|C_{XV_i}), \forall\ i$. Puisque X est parent de V_i, tous les arcs provenant des autres parents de V_i dans $\mathcal{G}'$ ont été coupés. Or, d'après l'algorithme de coupe, ce n'est possible que s'ils sont parents de X et de V_i. Par conséquent, les labels de C_{XV_i} sont inclus dans $\{X, Z_1, \ldots, Z_p\}$. S'il existe un Z_j n'appartenant pas à C_{XV_i}, alors ce dernier est indépendant de V_i conditionnellement à C_{XV_i} ; en effet, dans le cas contraire, il existerait une chaîne $\{U_1, \ldots, U_k\}$ reliant Z_j et V_i et ne passant par aucune variable de C_{XV_i}. D'après la propriété de d-connexion[7], cette chaîne ne doit avoir que des arcs non convergents. Par conséquent, puisque Z_j ne peut être un enfant de V_i (sinon on aurait un circuit), le dernier maillon de la chaîne est nécessairement un parent de V_i. C'est donc soit X, mais X appartient à C_{XV_i}, soit un parent U de V_i dont l'arc (U, V_i) a été coupé, ce qui implique que $U \in C_{XV_i}$. Dans les deux cas, Z_j serait indépendant de V_i conditionnellement à C_{XV_i}. En conclusion, $P(e^-_{XV_i}|C_{XV_i}) = P(e^-_{XV_i}|X, Z_1, \ldots, Z_p)$.

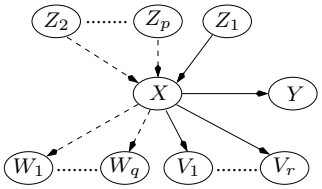


Figure 22. *Le cas général*

De même, quels que soient i et j, $e^-_{XV_i}$ est indépendante de $e^-_{XV_j}$ conditionnellement à $X, Z_1, \ldots, Z_p$. En effet, par hypothèse, chaque information e_{X_k} est indépendante du reste du réseau conditionnellement à X_k. Donc si, pour toutes les variables X_k, X_l telles que $e_{X_k} \in e^-_{XV_i}$ et $e_{X_l} \in e^-_{XV_j}$, on montre que X_k est indépendante de X_l conditionnellement à $X, Z_1, \ldots, Z_p$, alors $e^-_{XV_i}$ est aussi indépendante de $e^-_{XV_j}$ conditionnellement à $X, Z_1, \ldots, Z_p$. Faisons une démonstration par l'absurde : considérons que X_k est dépendante de X_l conditionnellement à $X, Z_1, \ldots, Z_p$, alors il existe une chaîne d-connectante entre X_k et X_l dans $\mathcal{G}'$ (avant les coupes). Notons que chaque fois qu'on a coupé un arc (A, B), il existait deux autres arcs (A, C) et (C, B) qui, eux, n'ont pas été coupés. Par conséquent, si la chaîne d-connectante entre X_k et X_l contient l'arc coupé, on peut remplacer cet arc par la séquence (A, C) et (C, B). On retrouve à nouveau une chaîne d-connectante, sauf si $C \in \{X, Z_1, \ldots, Z_p\}$. Si $C = X$, alors, comme A est parent de X, $A \in \{Z_1, \ldots, Z_p\}$, mais alors (A, B) ne peut faire partie de la chaîne avant coupe car elle passe par un arc sortant d'un Z_i. Si

7. Deux variables X et Y sont dépendantes si et seulement si elles sont d-connectées, c'est-à-dire qu'il existe une chaîne $\{A_1, \ldots, A_k\}$ telle que $A_1 = X$, $A_k = Y$, et telle que si la chaîne contient : i) des arcs (A_{k-1}, A_k) et (A_{k+1}, A_k), alors soit A_k soit un de ses descendants a reçu une information ; et ii) soit un arc (A_k, A_{k+1}), soit (A_{k+1}, A_k), mais pas les deux, alors A_k n'a pas été instancié.

C est un des Z_i, alors B est forcément égal à X, ce qui implique que A est un parent de X et que la chaîne avant coupe contenait déjà un arc sortant d'un Z_i. Elle ne pouvait donc pas être d-connectante. C'est impossible par hypothèse. Donc après coupe, la chaîne reliant X_k et X_l est forcément encore d-connectante. Or, lorsque toutes les coupes ont été effectuées, la chaîne reliant ces deux nœuds passe par les arcs (X, V_i) et (X, V_j). Elle n'est donc pas d-connectante. Par conséquent, il est impossible que X_k soit dépendante de X_l conditionnellement à $X, Z_1, \ldots, Z_p$.

Par conséquent,

$$\prod_{i=1}^{r} \lambda_X(V_i) = P\left(\bigcup_{i=1}^{r} e^-_{XV_i} | X, Z_1, \ldots, Z_p\right).$$

(Z_1, X) étant le seul arc restant après coupe, et provenant des parents de X, cela signifie que tous les autres arcs (Z_i, X) ont été coupés sur des triangles $Z_i Z_1 X$. Par conséquent, $\{Z_1, \ldots, Z_p\} \subseteq C_{Z_1X}$. Or C_{Z_1X} ne peut contenir aucun autre label puisque ceux-ci sont forcément des parents de X (d'après l'algorithme de coupe). Donc $\pi_{Z_1}(X) = P(e^+_{Z_1X}, C_{Z_1X}) = P(e^+_{Z_1X}, Z_1, \ldots, Z_p)$. Pour des raisons similaires à celles du paragraphe précédent, $e^+_{Z_1X}$ est forcément indépendant de X et de e_X conditionnellement à $\{Z_1, \ldots, Z_p\}$. Par conséquent, $P(X, e_X | Pa(X))\pi_{Z_1}(X) = P(X, e_X, e^+_{Z_1X} | Z_1, \ldots, Z_p)$. Pour les mêmes raisons, les $e^-_{XV_i}$ sont aussi indépendants de e_X et de $e^+_{Z_1X}$ conditionnellement à $\{X, Z_1, \ldots, Z_p\}$. Par conséquent,

$$P(X, e_X | Pa(X)) \times \pi_X(Z_1) \times \prod_{j=1}^{r} \lambda_{V_j}(X) = P(e^+_{XY}, X, Z_1, \ldots, Z_p).$$

Or par symétrie entre Y et les V_i, $C_{XY} \subseteq \{X, Z_1, \ldots, Z_p\}$; donc si l'on somme $P(e^+_{XY}, X, Z_1, \ldots, Z_p)$ sur toutes les variables n'appartenant pas à C_{XY}, on obtient $P(e^+_{XY}, C_{X,Y})$, soit précisément le message décrit dans l'énoncé de la proposition. La démonstration est similaire pour les messages remontant les arcs.

Pour finir, montrons que les complexités des calculs sont similaires chez Jensen et dans notre algorithme. On sait que la complexité chez Jensen est égale à la somme des tailles des cliques puisqu'on envoie autant de messages qu'il y a de liens dans l'arbre de jonction. On a vu que, dans notre algorithme et sur le graphe de la figure 15 (après coupe), les messages étaient :

$$\pi_Y(X) = \sum_{\mathcal{V} \setminus C_{XY}} \left[P(X, e_X | Pa(X)) P(e^+_{U_1X}, C_{U_1X}) \prod_{j=1}^{m} P(e^-_{XY_j} | C_{XY_j}) \right],$$

$$\lambda_Y(X) = \sum_{\mathcal{V} \setminus C_{XY}} \left[P(Y, e_Y | Pa(Y)) \prod_{j=1}^{p} P(e^-_{YS_j} | C_{YS_j}) \right],$$

car les nœuds du graphe ne peuvent avoir qu'un seul parent. On a vu dans les paragraphes précédents, que C_{XY} et les C_{XY_j} sont nécessairement inclus dans l'ensemble $\{X, \mathit{Pa}(X)\}$. Donc $\pi_Y(X)$ est égal à :

$$\sum_{\mathcal{V}\setminus C_{XY}} [P(X, e_X|\mathit{Pa}(X)) \prod_{j=1}^{m} P(e^-_{XY_j}|C_{XY_j}) \sum_{\mathcal{V}\setminus\{X,\mathit{Pa}(X)\}} P(e^+_{U_1X}, C_{U_1X}).$$

D'après le troisième paragraphe de la démonstration et l'algorithme de triangulation orientée, on sait que les labels correspondent aux cliques. Maintenant, les produits ci-dessus correspondent à un envoi de message dans le graphe orienté triangulé. Par conséquent, tout comme chez Jensen, la complexité de notre calcul correspond à la somme des cliques obtenues après chaque élimination. Nous en avons plus que Jensen dans la mesure où, après l'élimination d'un nœud X, une nouvelle clique peut être une sous-clique de celle obtenue pour X. Cela dit, dans ce cas, comme une variable aléatoire a au moins deux valeurs possibles, la taille de cette sous-clique est au moins deux fois plus petite que celle de la clique qui la contient. Par conséquent, on ne fait pas plus de deux fois les calculs de Jensen. D'où une complexité de calcul similaire à celle de Jensen. Cela dit, Jensen effectue autant de divisions que de multiplications, et nous n'en effectuons pas. ♦

8. Bibliographie

[BAR 94] BAR-YEHUDA R., GEIGER D., NAOR J., ROTH R., « Approximation algorithms for the vertex feedback set problem with application to constraints satisfaction and Bayesian inference », *Proceedings of the 5th annual ACM-SIAM Symposium on Discrete Algorithms*, 1994, p. 344-354.

[BEC 96a] BECKER A., GEIGER D., « Optimization of Pearl's method of conditioning and greedy-like approximation algorithm for the vertex feedback set problem », *Artificial Intelligence*, vol. 83, 1996, p. 167-188.

[BEC 96b] BECKER A., GEIGER D., « A sufficiently fast algorithm for finding close to optimal junction trees », *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 1996, p. 81-89.

[BEC 99a] BECKER A., BAR-YEHUDA R., GEIGER D., « Random algorithm for the Loop Cutset Problem », *proceedings of the fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999.

[BEC 99b] BECKER A., NAÏM P., *Les réseaux Bayesiens : modèles graphiques de connaissance*, Eyrolles, 1999.

[BRE 99] BREESE J., HECKERMAN D., « Decision-Theoretic Troubleshooting », *IEEE Transactions on Systems, Man & Cybernetics, Part A (Systems & Humans)*, vol. 26, n° 6, 1999, p. 838-42.

[CON 97] CONATI C., GERTNER A., VANLEHN K., DRUDZEL M. J., « On-Line Student Modeling for Coached Problem Solving Using Bayesian Networks », JAMESON A., PARIS C., TASSO C., Eds., *User Modeling : Proceedings of the Sixth Internatioonal Conference, UM'97*, Vienna, New York, 1997, Springer.

[COW 99] COWELL R., DAWID A., LAURITZEN S., SPIEGELHALTER D., *Probabilistic Networks and Expert Systems*, Springer-Verlag, 1999.

[DAR 95] DARWICHE A., « Conditioning Methods for Exact and Approximate Inference in Causal Networks », BESNARD P., HANKS S., Eds., *Proceedings of the eleventh Conference on Uncertainty in Artificial Intelligence*, 1995.

[DIE 96] DIEZ F., « Local Conditioning in Bayesian Networks », *Artificial Intelligence*, vol. 87, 1996, p. 1–20.

[FAÕ0] FAŸ A., JAFFRAY J.-Y., « A justification of local conditioning in Bayesian networks », *International Journal of Approximate Reasoning*, vol. 24, n° 1, 2000, p. 59–81.

[GON 03] GONZALES C., WUILLEMIN P.-H., « A Qualitative Scheme for Optimized propagation in Bayesian Networks », rapport, 2003, LIP6 – université Paris 6.

[HOR 98] HORVITZ E., BREESE J., HECKERMAN D., HOVEL D., ROMMELSE K., « The Lumiere Project : Bayesian User Modeling for Inferring the Goals and Needs of Software Users », *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 1998, p. 256-265.

[JEN 90] JENSEN F., LAURITZEN S., OLESEN K., « Bayesian Updating in Causal Probabilistic Networks by Local Computations », *Computational Statistics Quarterly*, vol. 4, 1990, p. 269-282.

[JEN 94] JENSEN F., JENSEN F., « Optimal junction trees », MANTARAS R., POOLE D., Eds., *Proceedings of the Tenth Conference on Uncertainty in Articial Intelligence*, Morgan Kaufmann, 1994, p. 360-366.

[JEN 96] JENSEN F., *An Introduction to Bayesian Networks*, Springer-Verlag, North America, 1996.

[KJÆ 90] KJÆRULFF U., « Triangulation of Graphs — Algorithms Giving Small Total State Space », rapport n° R-90-09, 1990, Dept. of Mathematics and Computer Science, Aalborg University.

[LAU 88] LAURITZEN S., SPIEGELHALTER D., « Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems », *The Journal of The Royal Statistical Society – Series B (Methodological)*, vol. 50, n° 2, 1988, p. 157-224.

[LAU 96] LAURITZEN S., *Graphical Models*, Clarendon Press, Oxford, United Kingdom, 1996.

[MAD 99] MADSEN A., JENSEN F., « LAZY propagation : A junction tree inference algorithm based on lazy inference », *Artificial Intelligence*, vol. 113, n° 1–2, 1999, p. 203–245.

[MIS 96] MISLEVY R., GITOMER D., « The Role of Probability-Based Inference in an Intelligent Tutoring System », *User Modeling and User-Adapted Interaction*, vol. 5, 1996, p. 253-282.

[NIE 00] NIELSEN T., WUILLEMIN P.-H., JENSEN F., KJÆRULFF U., « Using ROBDDs for inference in Bayesian networks with troubleshooting as an example », *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, 2000.

[PEA 88] PEARL J., *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference*, Morgan Kaufman Publishers, inc, 1988.

[PEO 91] PEOT M. A., SHACHTER R. D., « Fusion and Propagation with Multiple Observations in Belief Networks », *Artificial Intelligence*, vol. 48, 1991, p. 299-318.

[ROS 70] ROSE D., « Triangulated Graphs and the Elimination Process », *Journal of Mathematical Analysis and Applications*, vol. 32, 1970, p. 597-609.

[SHA 86] SHACHTER R., « Evaluating Influence Diagrams », *Operations Research*, vol. 34, n° 6, 1986, p. 871-882.

[SHA 90] SHAFER G., SHENOY P., « Probability propagation », *Annals of Mathematics and Artificial Intelligence*, vol. 2, 1990, p. 327–352.

[SHA 94] SHACHTER R. D., ANDERSEN S. K., SZOLOVITS P., « Global Conditioning for Probabilistic Inference in Belief Networks », LOPEZ DE MANTARAS R., POOLE D., Eds., *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, 1994.

[SHE 90] SHENOY P., SHAFER G., « Axioms for Probability and Belief-function Propagation », SHACHTER R., LEVITT T., LEMMER J., KANAL L., Eds., *Proceedings of Uncertainty in Artificial Intelligence*, 1990, p. 169-198.

[SHO 76] SHORTLIFFE E., *Computer-Based Medical Consultation : MYCIN*, Elsevier, 1976.

[SHO 97] SHOIKHET K., GEIGER D., « Finding Optimal Triangulations via Minimal Vertex Separators », *Proceedings of AAAI-97*, 1997.

[SUE 90] SUERMONDT H., COOPER G., « Probabilistic Inference in Multiply connected Belief Networks Using Loop Cutsets », *International journal of approximate reasoning*, vol. 4, 1990, p. 283-306.

[YAN 81] YANNAKAKIS M., « Computing the Minimum Fill-in is NP-complete », *SIAM journal on Algebraic and Discrete Methods*, vol. 2, 1981, p. 77-79.