Inférence incrémentale pour les modèles probabilistes relationnels et application aux systèmes à base de règles orientés objet

Hamza Agli [1], **Philippe Bonnard** [1], **Christophe Gonzales** [2], **Pierre-Henri Wuillemin** [2]

1. *IBM France Lab, 9 rue Verdun, Gentilly, France*
   {hamza.agli,philippe.bonnard}@fr.ibm.com
2. *Sorbonne Université, CNRS, LIP6, UMR 7606, F-75005 Paris, France*
   {christophe.gonzales,pierre-henri.wuillemin}@lip6.fr

RÉSUMÉ. *Cet article s'intéresse à l'exploitation de règles probabilistes dans les systèmes à base de règles métier orientés objet (OO-BRMS). Afin de faciliter la modélisation des distributions de probabilités dans ces systèmes, nous proposons d'utiliser les modèles probabilistes relationnels (PRM), qui sont une extension orientée objet des réseaux bayésiens. Lors de l'exploitation des OO-BRMS, les requêtes adressées aux PRM sont nombreuses et les réponses doivent être calculées rapidement. Pour cela, nous proposons un nouvel algorithme tirant parti des spécificités des OO-BRMS : i) tout d'abord les probabilités à calculer ne concernent que des sous-ensembles de toutes les variables aléatoires des PRM, ce que l'on appelle des cibles ; ii) les requêtes successives diffèrent peu les unes des autres. Notre nouvel algorithme, IJTI, exploite ces deux spécificités afin d'optimiser les calculs. Nous prouvons mathématiquement que notre approche fournit des résultats exacts et montrons son efficacité par des résultats expérimentaux.*

ABSTRACT. *This article investigates the exploitation of probabilistic rules within object-oriented business rule management systems (OO-BRMS). In order to facilitate the modelling of the probability distributions in these systems, we propose to use probabilistic relational models (PRM), which are object-oriented extensions of Bayesian networks. When OO-BRMS are exploited by users, numerous requests are sent to the PRM and their answers need be computed very quickly. For this purpose, we propose a novel algorithm that exploits the specificities of OO-BRMS: i) first, the probabilities of interest concern only a subset of the PRM's random variables, which are called targets; and ii) successive requests differ only slightly. Our new algorithm, IJTI, exploits these two specificites in order to optimize computations.*

MOTS-CLÉS : *systèmes à base de règles, réseaux bayésiens, inférence incrémentale.*

KEYWORDS: *rule based systems, Bayesian networks, incremental inference.*

DOI:10.3166/RIA.32.111-132

1. Introduction

Les systèmes de gestion de règles métier orientés objets (OO-BRMS) sont des plates-formes applicatives complexes qui fournissent des outils pour automatiser les prises de décision métier. Ils sont populaires, en particulier parce qu'ils sont considérés comme l'évolution par excellence des systèmes experts à base de règles. En distinguant l'application informatique de la logique métier, ces systèmes facilitent l'écriture, la vérification et le déploiement des politiques métiers complexes des entreprises. En effet, dans de tels systèmes, les experts métiers et les spécialistes en informatique peuvent collaborer de manière relativement indépendante (Berstel-Da Silva, 2014 ; Graham, 2007). Malheureusement, si les BRMS sont bien adaptés pour la prise de décision dans le certain, c'est-à-dire lorsque les données sur lesquelles reposent les décisions sont complètes et permettent d'inférer les décisions optimales via des inférences logiques classiques, les BRMS rencontrent des difficultés pour tenir compte des incertitudes résultant de données incomplètement connues. De manière générale, la question d'incertitude dans les systèmes à base de règles a été traitée dans la littérature selon trois approches :

– Certains systèmes utilisent des modèles heuristiques affectant aux règles des poids représentant leurs degrés de vérité. C'est par exemple le cas des facteurs de certitude ou bien des ratios de vraisemblance (Buchanan, Shortliffe, 1984 ; Hart *et al.*, 1977). Malheureusement, ces modèles souffrent de limitations pratiques et théoriques (Heckerman, Shortliffe, 1992) que l'on pourrait pallier par l'exploitation de modèles probabilistes. Leurs performances peuvent en être significativement affectées (Ng, Abramson, 1990).

– Certains systèmes ont recours à la *logique floue* (Zadeh, 1965) afin de capturer les imprécisions sur des données d'entrée/sortie du système en terme d'une logique multi-valuée. Mais, là encore, ce type de modèles n'est pas complètement adapté aux enchainements multiples d'inférence fréquemment utilisé dans les BRMS et peuvent engendrer des conclusions incohérentes (Elkan, 1993).

– Enfin, il existe des systèmes reposant sur des techniques probabilistes, en particulier des réseaux bayésiens (BN) (Pearl, 1988 ; Koller, Friedman, 2009). Un BN est une représentation graphique compacte d'une distribution de probabilité. On peut le voir comme une base de connaissance probabiliste dans laquelle on effectue des requêtes *probabilistes*. Les BN sont largement utilisés dans des applications telles que le diagnostic médical (Beinlich *et al.*, 1989), la gestion du risque et les systèmes d'aide à la décision (Naim *et al.*, 2007).

Dans cet article, nous nous focalisons sur ce dernier type de systèmes et proposons un nouveau modèle permettant de passer à l'échelle, que ce soit en termes de modélisation mais également en termes de temps de calcul nécessaire à la prise de décision. Plus exactement, plutôt que de décrire les règles métier avec des distributions de probabilités *isolées* et de n'exploiter les réseaux bayésiens que pour obtenir des inférences rapides, nous proposons de coupler les BRMS avec les modèles probabilistes relationnels (PRM) (Koller, Pfeffer, 1998 ; Pfeffer, 2000 ; Torti *et al.*, 2013). Ces derniers sont une extension objet des BN, qui permettent de modéliser et maintenir

des distributions de probabilité de grandes tailles. Leurs paradigmes objet et relationnel sont tout à fait en accord avec les OO-BRMS, ce qui fait des PRM un modèle bien adapté pour modéliser les incertitudes au sein des OO-BRMS. L'idée clef réside dans le fait d'apparier des objets PRM représentant les incertitudes avec les objets BRMS représentant les règles sur lesquelles portent ces incertitudes. Ainsi, la modélisation et la maintenance de règles probabilistes en est-elle fortement facilitée. Si cela permet de créer aisément des OO-BRMS "probabilistes" de grandes tailles, il reste à établir des procédures efficaces afin de répondre aux requêtes adressées au système. Pour cela, il convient de réaliser rapidement des inférences dans les PRM associés aux BRMS. Dans cet article, nous proposons de transformer les PRM en BN, ce que l'on appelle des réseaux bayésiens "*groundés*", et d'appliquer dans ces BN un algorithme d'inférence efficace de la littérature, comme par exemple Lazy Propagation (Madsen, Jensen, 1999). Afin de passer à l'échelle, nous proposons de tenir compte du contexte dans lequel ces inférences sont réalisées : en effet, dans les BRMS, i) seules les probabilités d'un sous-ensemble de toutes les variables aléatoires sont utiles pour la prise de décision, c'est ce que l'on appelle les *variables cibles*; et ii) les requêtes successives adressées au système pour calculer ces probabilités ont souvent étroitement liées et on peut tirer parti des calculs réalisés dans les inférences précédentes. Nous proposons donc ici un nouvel algorithme incrémental orienté "cibles" dédié à cette tâche et fondé sur des envois de messages au sein d'un arbre de jonction (JT). L'idée clé de notre algorithme, appelé « Inférence d'Arbre de Jonction Incrémentale » (*IJTI* [1]), consiste à limiter les calculs seulement aux parties du JT qui sont pertinentes pour les *cibles* et qui ont été invalidées par les changements incrémentaux. En conséquence, *IJTI* optimise les calculs d'inférence probabiliste.

Cet article est organisé comme suit. Dans la section suivante, nous décrivons comment l'on peut apparier des PRM dans les OO-BRMS. Les sections suivantes sont consacrées à notre algorithme d'inférence incrémentale. Ainsi, dans la section 3, nous présentons les techniques d'inférence dans les BN ainsi que les différents types d'incrémentalité. Dans la section 4, nous présentons notre nouvel algorithme d'inférence incrémentale et nous justifions mathématiquement sa justesse. Les démonstrations sont fournies en appendice. Ensuite, nous mettons expérimentalement en évidence l'efficacité de notre contribution. Enfin, quelques conclusions et perspectives sont fournies dans la section 6.

2. BRMS et PRM

Dans cette section, nous décrivons les systèmes de règles métiers et expliquons comment les coupler avec les PRM.

Les systèmes à base de règles sont constitués de règles de la forme « *SI condition ALORS action* » qui sont elles-mêmes exploitées dans des algorithmes de raisonnement par chaînage avant. Les OO-BRMS conjuguent ces systèmes à base de règles

1. Incremental Junction Tree Inference.

avec un modèle orienté objet. L'exemple jouet suivant va nous permettre d'illustrer ces concepts. Supposons que l'on s'intéresse à la détection de fraude à l'assurance maladie et que notre modèle soit constitué de trois classes représentant respectivement un professionnel de santé, un patient de ce professionnel et une demande de remboursement de soins de la part de ce patient. La figure 1 montre un diagramme UML d'une telle application. Dans celle-ci, on souhaite vérifier, en utilisant un système à base de règles, si le professionnel de santé a émis une ordonnance frauduleuse. Ainsi, en exécutant un ensemble de règles couplé avec une heuristique de score, on peut détecter des anomalies, celles-ci correspondant très probablement à des fraudes. Par exemple, la règle 1 générique ci-dessous indique qu'il faut lever une alerte dès lors qu'un professionnel de santé (quel qu'il soit) facture des soins trop élevés à l'un de ses patients. Dans un contexte orienté objet, une telle règle est encodée comme une classe et l'on exploite toutes les instances de cette classe dans la mémoire de travail ("*working memory*").

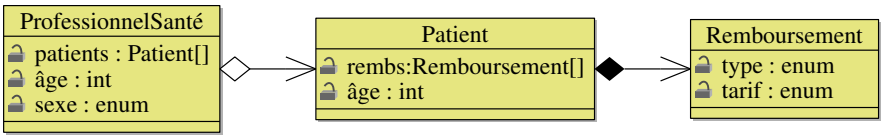


Figure 1. Diagramme UML simplifié d'une application de détection de fraude à l'assurance maladie

Règle 1 – détection de tarifs excessifs

```
SI hp est de type ProfessionnelSanté &
   pat est de type Patient &
   remb est de type Remboursement &
   pat appartient à hp.patients &
   remb appartient à pat.rembs &
   remb.tarif == élevé
ALORS LeverAlerteTarifExcessif (hp)
```

Lorsque les préconditions des règles sont complètement observées et ce de manière parfaite, c'est-à-dire sans imprécisions, et lorsque l'on peut inférer les conséquences de ces préconditions à l'aide de paradigmes logiques, comme c'est le cas dans la règle ci-dessus, les systèmes à base de règles se révèlent particulièrement efficaces. Les règles sont alors manipulées dans des algorithmes de type "*recherche de motifs*", par exemple des RETE étendus (Forgy, 1982). Cependant, lorsque les conséquences ne peuvent être inférées qu'avec un certain degré de certitude ou bien lorsque certaines préconditions son non observées ou bien imparfaitement observées, ces systèmes doivent être couplés avec des modèles d'incertitudes expressifs et permettant des calculs efficaces. C'est le cas de la règle 2 ci-dessous :

Règle 2 – détection de fraude aux tarifs excessifs

```
SI hp est de type ProfessionnelSanté &
   pat est de type Patient &
   remb est de type Remboursement &
```

```
   pat appartient à hp.patients &
   remb appartient à pat.rembs &
   remb.tarif == élevé
ALORS Proba ( Fraude(hp) ) >=  70%
```

Cependant, en pratique, il est rare que les règles soient toutes indépendantes les unes des autres. Aussi, il convient de tenir compte des dépendances probabilistes afin de calculer correctement les probabilités nécessaires à la prise de décision. Les réseaux bayésiens peuvent ainsi servir pour modéliser l'ensemble des relations probabilistes décrites dans les règles. Cela dit, dans les OO-BRMS, les règles sont encodées à partir de modèles objets, et il peut être plus judicieux d'utiliser également un modèle objet pour encoder les distributions de probabilités. Les PRM (probabilistic relational models) sont donc adaptés à cette tâche (Torti *et al.*, 2010 ; Gonzales, Wuillemin, 2011 ; Torti, 2012 ; Torti *et al.*, 2013). Sans rentrer dans les détails, l'idée est la suivante : dans un OO-BRMS, on définit une classe générique de patient, que l'on instancie dans la mémoire de travail avec autant de patients que nécessaire. Dans un PRM, on procède de la même manière: on définit des classes de distributions de probabilités conditionnelles multivariées et on utilise leurs instances pour former la distribution jointe (autrement dit le réseau bayésien) dont on a besoin. Ce dernier s'appelle alors un réseau bayésien « *groundé* » et il est donc formé par une répétition de « motifs » probabilistes qui correspondent à des distributions conditionnelles. Par exemple, le PRM correspondant à l'OO-BRMS de la figure 1 pourrait être celui de la figure 2. Ici les classes sont symbolisées par les grands rectangles. Elles correspondent préci-

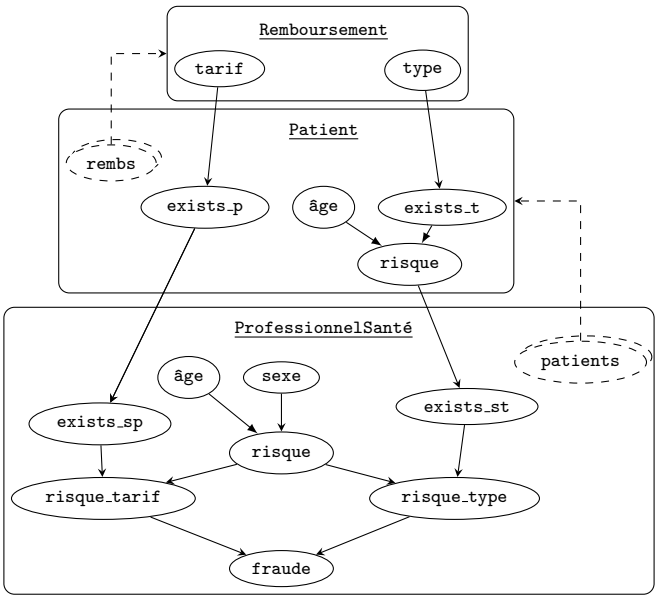


Figure 2. Schéma possible de classes PRM pour la détection de fraude

sément aux classes du diagramme UML. À l'intérieur des rectangles, les ellipses en

traits pleins correspondent à des attributs, c'est-à-dire à des « variables aléatoires génériques ». Les arcs représentent les dépendances probabilistes entre ces attributs et, à chaque attribut est associé sa table de probabilité conditionnellement à ses parents. Lorsque l'on instancie les classes, il suffit de recopier tous ces attributs pour former des variables aléatoires, ainsi que leurs arcs et leurs tables de probabilité conditionnelle afin de créer un véritable réseau bayésien. Par exemple, si l'on considère un problème à trois patients, il suffit de recopier trois fois le sous-graphe correspondant à la classe `Patient`. Certaines classes ont des tables de probabilités dépendant d'attributs d'autres classes. C'est le cas par exemple des remboursements d'un patient, qui dépendent de l'attribut `tarif` de la classe `Remboursement`. Dans ce cas, les PRM utilisent un mécanisme dit de « slot de référence » afin de permettre de référencer des attributs d'autres classes. Celui-ci est symbolisé par des doubles ellipses en pointillés. Dans l'exemple de la figure 2, l'attribut `rembs` de la classe `Patient` est un slot de référence multiple qui pointe vers la classe `Remboursement`.
Les PRM sont également dotés de ce que l'on appelle des « agrégateurs ». Ces derniers permettent de faire en sorte que le nombre de parents d'un attribut, une fois instancié, n'est pas fixé à l'avance dans la classe. Cela permet par exemple à un professionnel de santé d'avoir un nombre de patients différent de celui de ses confrères. Dans ce cas, dans la classe, la probabilité conditionnelle de l'attribut est encodée à l'aide d'une fonction et non d'une table de taille fixe. L'agrégateur `exist_st` de la classe `ProfessionnelSanté` illustre cette notion. Il sert à évaluer notre croyance sur l'existence d'un patient, parmi tous les patients du professionnel en question, ayant un risque élevé et lié au type du remboursement demandé. Pour déterminer s'il y a eu fraude, on considère l'ensemble des instances de la classe `Patient`, on détermine ceux pour lesquels au moins un des remboursements dans `rembs` est de `tarif` élevé. Ensuite, on détermine les instances de `ProfessionnelSanté` liés à de tels patients. Toutes ces instances forment un réseau bayésien dans lequel une simple inférence permet de calculer la probabilité qu'il y ait eu fraude. Notons que, lorsque l'on réalise l'inférence dans ce réseau bayésien dit "groundé", la présence d'agrégateurs n'augmente pas la complexité de cette inférence. En effet, dans ce cas, l'agrégateur peut être simplement encodé comme une table de probabilité conditionnelle classique. En revanche, si l'on exploite des mécanismes d'inférence structurée (Wuillemin, Torti, 2012), la présence d'agrégateurs peut induire une augmentation de la complexité dans la mesure où des instances différant par le nombre de parents de leurs agrégateurs doivent être considérées comme structurellement différentes et ne peuvent donc pas donner lieu à des optimisations des calculs.

L'inconvénient de ces mécanismes d'inférence réside dans le fait que, dans un OOBRMS, on est amené à effectuer très souvent de telles inférences. Or, celles-ci peuvent être coûteuses en temps de calcul. Fort heureusement, ces inférences sont souvent très corrélées, c'est-à-dire que le réseau bayésien employé dans une inférence varie très peu de celui employé lors de l'inférence précédente. On peut donc espérer gagner des temps de calculs significatifs si l'on exploite intelligemment ces redondances. Dans les sections suivantes, nous développons un tel mécanisme.

3. Calculs dans les BN et incrémentalité

En pratique, plusieurs types de requêtes sont employées dans les BN. On peut par exemple citer l'explication la plus probable (MPE), qui consiste à déterminer l'ensemble des valeurs de toutes les variables aléatoires ayant la probabilité jointe *a posteriori* la plus élevée (étant donné un ensemble d'observations), ou bien la requête MAP (maximum *a posteriori*), qui consiste à déterminer les valeurs d'un sous-ensemble fixé de variables dont la loi jointe *a posteriori* est maximale [2]. Dans cet article, nous nous intéressons à un troisième type de requête, d'usage courant dans les BRMS : le calcul de probabilités marginales *a posteriori* de certaines variables aléatoires (*cibles*) étant donné un ensemble d'observations. C'est ce que l'on appelle communément en anglais des requêtes d'*inférence*. Ici, les observations peuvent être certaines, c'est-à-dire que l'on connaît précisément la valeur de certaines variables aléatoires. On parle alors d'observations *dures* (*hard evidence* en anglais). Mais elles peuvent être également *souples* (*soft evidence* en anglais), c'est-à-dire qu'elles ne permettent pas de déterminer précisément la valeur des variables aléatoires sur lesquelles elles portent. Par exemple, on peut observer qu'une température est inférieure à 20°C sans pour autant savoir précisément quelle est cette température. Bien que l'inférence soit un problème PP-complet dans le cas général (Darwiche, 2009), plusieurs algorithmes d'inférence exacts ou approchés existent dans la littérature (Lauritzen, Spiegelhalter, 1988 ; Madsen, Jensen, 1999 ; Shenoy, Shafer, 2008 ; Allen, Darwiche, 2003 ; Bacchus *et al.*, 2003 ; Sang *et al.*, 2005 ; Chavira, Darwiche, 2008). Des extensions pour manipuler les systèmes de grandes tailles ont également été proposées (Koller, Pfeffer, 1998 ; Pfeffer, 2000 ; Torti *et al.*, 2013), ainsi que des méthodes pour tenir compte d'aspects temporels (Dean, Kanazawa, 1989 ; Murphy, 2002 ; Robinson, Hartemink, 2009 ; Dubuisson *et al.*, 2012). La complexité des BN n'a cessé de grandir et les algorithmes d'inférence sont de plus en plus performants. Malgré cela, l'exploitation d'incrémentalités reste encore marginale dans la littérature sur l'inférence. Dans ce article, nous proposons donc un algorithme original exploitant cette propriété ainsi que le fait de ne s'intéresser qu'à un ensemble potentiellement restreint de *cibles*, c'est-à-dire de variables dont on souhaite connaître les probabilités marginales *a posteriori*.

Les mécanismes efficaces d'inférence exploitent souvent des structures graphiques particulières construites à partir du BN initial. C'est le cas notamment des algorithmes par envoi de messages dans les arbres de jonction (JT, *Junction Tree* en anglais) (Jensen *et al.*, 1990 ; Shenoy, Shafer, 2008 ; Madsen, Jensen, 1999). Ils semblent être de bons candidats pour réaliser des inférences incrémentales dans les BN et nous allons donc fonder notre algorithme sur cette technologie. Certains algorithmes exploitent déjà en partie l'incrémentalité (Madsen, Jensen, 1999), mais le résultat est loin d'être optimal en temps de calcul lorsque les *cibles* changent dynamiquement et forment un sous-ensemble strict de l'ensemble des variables aléatoires. La situation est pire lorsque la structure du JT évolue elle-même dans le temps. C'est une probléma-

2. Répondre à une requête MAP consiste donc à marginaliser les variables qui sont hors du sous-ensemble puis à effectuer un MPE sur la distribution résultante.

tique pour les OO-BRMS, où les changements de la structure du BN, des observations et des *cibles* sont fréquents.

Dans (D'Ambrosio, 1993), quatre critères d'incrémentalité relatifs à l'inférence probabiliste ont été présentés : incrémentalité par rapport aux ressources de calculs, aux requêtes, aux observations et au modèle graphique. Dans cet article, on s'intéresse à tous ces critères, particulièrement aux trois derniers. Étonnamment, très peu d'algorithmes d'inférence abordent tous ces aspects. Ainsi, Darwiche (1998) traite l'incrémentalité du point de vue des requêtes en reconfigurant dynamiquement certains JT quand ces requêtes changent. Mais la structure du BN est supposée statique, ce qui n'est pas forcément le cas dans les BRMS. Lin et Druzdzel (1998), exploitent un raisonnement fondé sur la *pertinence* pour identifier les parties du réseau qui sont pertinentes pour les calculs, puis ils mettent à jour plusieurs sous-réseaux dont l'union couvre le réseau d'origine. Cet algorithme ne prend pas en compte les calculs effectués précédemment. Il est de même pour l'algorithme BayesBall (Shachter, 1998) qui détermine l'information requise et l'information non pertinente pour élaguer le BN initial avant de procéder au calcul des probabilités conditionnelles. Dans (Madsen, Jensen, 1999), un algorithme d'inférence par JT incrémental a été proposé. Il exploite les indépendances induites par les mises à jour des observations incrémentales. La structure du JT est toutefois statique et tous les nœuds sont considérés comme *cibles*, ce qui n'est pas optimal dans notre contexte. Dans (Flores *et al*., 2003), seule l'incrémentalité de la structure du JT est prise en compte, mais pas celle des requêtes, et les calculs probabilistes précédents ne sont pas non plus exploités. Dans la même perspective, Li et al. (2006) proposent une compilation du *BN* d'origine en une forme normale conjonctive couplée avec des techniques de cache, ce qui améliore l'inférence quand la structure du réseau est mise à jour. Cette méthode ne prend pas en considération de façon optimale les requêtes et les évidences. Dans la section suivante, nous proposons un nouvel algorithme permettant de pallier tous ces problèmes.

4. Optimisation de l'inférence incrémentale : *IJTI*

4.1. Notations et préliminaires

Un BN est représenté par un couple $(\mathcal{G}, \Theta)$, où $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ est un graphe orienté sans circuit (DAG [3]). $\mathcal{V}$ est un ensemble de nœuds représentant des variables aléatoires [4]. $\mathcal{A}$ est un ensemble d'arcs et $\Theta = \{P(X|\text{Pa}(X)) : X \in \mathcal{V}\}$ est l'ensemble des tables de probabilités (CPT) des variables de $\mathcal{V}$ conditionnellement à leurs parents dans $\mathcal{G}$. Le BN encode la loi de probabilité jointe sur $\mathcal{V}$ comme le produit de ces CPT. Dans cet article, l'inférence probabiliste est fondée sur un algorithme par envoi de messages dans le JT. La construction de ce dernier consiste d'abord à convertir le DAG $\mathcal{G}$ en un graphe non orienté en ajoutant, pour chaque nœud dans $\mathcal{V}$, des arêtes entre tous ses parents (*moralisation*), en enlevant les orientations des arcs restants et

3. Directed Acyclic Graph.

4. Par abus, nous ne faisons pas de distinction entre les nœuds et les variables aléatoires qu'ils représentent.

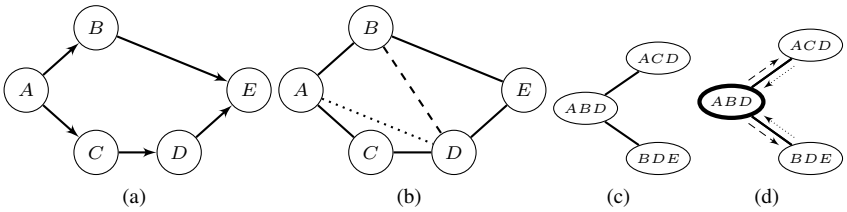


Figure 3. Construction d'un JT

en ajoutant par la suite des arêtes de telle sorte que, dans tout cycle d'au moins quatre nœuds, deux nœuds non adjacents sont reliés entre eux (*triangulation*). Les nœuds du JT correspondent alors à des sous-graphes complets et maximaux (des cliques) du graphe résultant. Ces cliques sont reliées par des arêtes de telle sorte que : $i)$ le JT ne contient aucun cycle ; et $ii)$ tout couple de cliques d'intersection non vide sont reliées par une chaîne, sur laquelle toutes les cliques contiennent cette intersection.

La figure 3a montre un exemple d'un DAG, son graphe moral et triangulé est donné dans la figure 3b, où les arêtes avec des traits discontinus et celles en pointillés représentent les arêtes introduites respectivement pendant la *moralisation* et la *triangulation*. Enfin, la figure 3c est une représentation possible du JT correspondant. Notons qu'un JT peut être une forêt, par exemple, lorsque le DAG $\mathcal{G}$ n'est pas connexe. Dans notre approche, traiter une forêt est équivalent à réitérer le même processus sur ses composantes connexes. D'où, sans perte de généralité, nous considérerons dans la suite que le JT sur lequel nous travaillons est un arbre $\mathcal{T}$. Dans ce qui suit, pour n'importe quel JT $\mathcal{T}$, nous noterons $\mathcal{V}(\mathcal{T})$ et $\mathcal{E}(\mathcal{T})$ l'ensemble de ses cliques et de ses arêtes respectivement. Les algorithmes par envoi de messages consistent à appliquer une *collecte* et une *distribution* à partir d'une racine prédéterminée $r \in \mathcal{V}(\mathcal{T})$. Pendant la *collecte*, les messages sont envoyés à partir des feuilles vers r et, pendant la *distribution*, ils sont envoyés dans la direction opposée. Pour garantir la justesse des calculs, pour toute arête $(i, j) \in \mathcal{E}(\mathcal{T})$, le message envoyé de i à j, noté $\psi_{i \to j}$, est calculé seulement quand la clique i a reçu les messages de tous ses voisins sauf j. La figure 3d montre un exemple de passage de messages avec $r = ABD$ (la clique en traits épais), les arcs en pointillés et en traits discontinus représentant respectivement les messages de *collecte* et de *distribution*. Le calcul de ces messages n'est pas le sujet de cet article, mais peut être trouvé dans (Madsen, Jensen, 1999). Toutefois, il est important de se rappeler que la complexité de l'algorithme d'inférence réside principalement dans le calcul de ces messages (multi-dimensionnels) [5]. Il est donc pertinent de s'intéresser à la minimisation du nombre de calculs de tels messages.

Dans un environnement incrémental, tous les messages n'ont pas besoin d'être recalculés chaque fois qu'une modification arrive, parce que, en pratique, beaucoup

5. La complexité en temps et en espace de l'inférence est exponentielle en la taille de la plus grande clique du JT moins un, ce que l'on appelle la *treewidth*.

resteront inchangés. Comme nous le verrons plus tard, avec l'utilisation des définitions suivantes, nous pouvons caractériser précisément ceux qui ont besoin d'une mise à jour due à l'ajout de nouvelles observations, des changements de structure ou/et des *cibles*.

DÉFINITION 1 (Chaîne). — *Soit* $\mathcal{T} = (\mathcal{V}(\mathcal{T}), \mathcal{E}(\mathcal{T}))$ *un JT.* $i_1, \ldots, i_{n+1}$ *est appelé une chaîne dans* $\mathcal{T}$ *si* $(i_\alpha, i_{\alpha+1}) \in \mathcal{E}(\mathcal{T})$ *pour tout* $\alpha \in \{1, \ldots, n\}$. *Pour simplifier, cette chaîne sera notée* $i_1 - i_{n+1}$ *et sa longueur* $\mathrm{len}(i_1 - i_{n+1})$ *(qui est ici égale à* n*).*

DÉFINITION 2 (Adjacence). — *Soit* $i, j \in \mathcal{V}(\mathcal{T})$, $i \neq j$. i *et* j *sont adjacents dans* $\mathcal{T}$ *ssi* $(i, j) \in \mathcal{E}(\mathcal{T})$. *L'ensemble des cliques adjacentes à* i *est noté* $\mathrm{Adj}(i)$, *c-à-d.,* $\mathrm{Adj}(i) := \{k \in \mathcal{V}(\mathcal{T}) : (i, k) \in \mathcal{E}(\mathcal{T})\}$. *Soit* $r \in \mathcal{V}(\mathcal{T})$, $r \neq i$, *alors* $\mathrm{Adj}_r(i)$ *dénote le singleton contenant la clique adjacente à* i *située sur la chaîne reliant* i *et* r, *c-à-d.,* $\mathrm{Adj}_r(i) := \{k \in \mathrm{Adj}(i) : k \in i - r\}$. *On définit aussi* $\mathrm{Adj}_r(r) := \emptyset$. *Finalement, soit* $\mathrm{Adj}_{\text{-}j}(i) := \mathrm{Adj}(i) \setminus \{j\}$ *l'ensemble des cliques adjacentes à* i *sauf* j.

Par exemple, dans la figure 4, $\mathrm{Adj}(i) = \{j, k_3, k_4\}$. De même, $\mathrm{Adj}_r(i) = \{k_3\}$ car k_3 est la seule clique se trouvant sur la chaîne entre i et r, et $\mathrm{Adj}_r(k_3) = \{r\}$.

On notera $\mathcal{V}_{\text{-}j}(i)$ l'ensemble des nœuds du sous-arbre maximal dans $\mathcal{T}$ qui contient i mais pas $\mathrm{Adj}_j(i)$. Sur la figure 4, cela correspond à l'ensemble des cliques dans le rectangle en traits discontinus. En effet, $\mathrm{Adj}_j(i)$ est le singleton contenant la clique adjacente à i se trouvant sur la chaîne $i - j$, autrement dit $\mathrm{Adj}_j(i) = \{j\}$. Dans l'arbre de jonction de la figure 4, le sous-arbre maximal contenant i mais pas j est bien constitué des cliques du rectangle en traits discontinus. Enfin, on notera $\mathcal{V}_j(i) = \mathcal{V}_{\text{-}j}(i) \cup \{j\}$. Au rectangle en traits discontinus, il faut donc rajouter la clique j, ce qui nous donne la partie grisée de la figure 4.

4.2. Caractérisation des messages

Un message $\psi_{i \to j}$ transitant dans $\mathcal{T}$ est orienté par définition. Il propage vers j (et, par induction, vers $\mathcal{V}_{\text{-}i}(j)$, l'ellipse en pointillés de la figure 4) toutes les informations pertinentes provenant des cliques dans $\mathcal{V}_{\text{-}j}(i)$ (le rectangle en traits discontinus de la figure 4), notamment toutes les observations reçues. Par conséquent, $\psi_{i \to j}$ a besoin d'être recalculé dès lors qu'une nouvelle observation ou des changements de structure

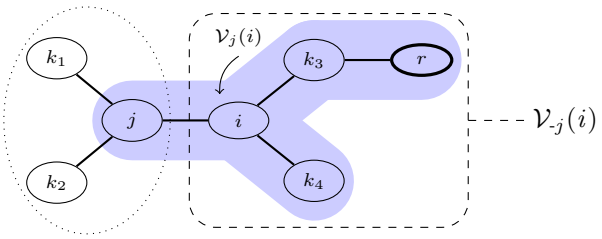


Figure 4. Les adjacences et sous-arbres dans un arbre de jonction

apparaissent dans $\mathcal{V}_{\text{-}j}(i)$ [6]. Cependant, même si $\mathcal{V}_{\text{-}j}(i)$ a reçu des observations, $\psi_{i\to j}$ n'a pas pas besoin d'être recalculé si $\mathcal{V}_{\text{-}i}(j)$ ne contient aucune *cible*. Dans ce cas, l'état de $\psi_{i\to j}$ devient « invalide » puisque son contenu est maintenant incorrect. Ceci ne pose pas de problème pour l'inférence actuelle, mais, pour les futures, nous devons prendre en compte cet état pour recalculer $\psi_{i\to j}$ s'il doit être utilisé par la suite.

Nous proposons donc ici un algorithme permettant d'identifier les messages à recalculer.

Soit $\mathcal{A}(\mathcal{T})$ l'ensemble de tous les arcs obtenus de $\mathcal{E}(\mathcal{T})$ en considérant toutes les orientations possibles, c'est-à-dire $\mathcal{A}(\mathcal{T}) := \bigcup_{(i,j)\in\mathcal{E}(\mathcal{T})}\{(i,j)\}\cup\{(j,i)\}$. Pour formaliser les conditions ci-dessus, nous commençons par caractériser les informations qui sont « locales » à i et j par :

DÉFINITION 3 (Label-message local λ). — $\lambda : \mathcal{A}(\mathcal{T}) \mapsto 2^{\{\epsilon,T\}}$ *est une fonction telle que :*

$$(i,j) \longmapsto \lambda_{i\to j} := \begin{cases} \{\epsilon\} & \textit{si l'état de } \psi_{i\to j} \textit{ est « invalide » ou si de nouvelles observations ou des changements de structure affectent } i \quad (1) \\ \{T\} & \textit{si } i \textit{ contient des} \text{ cibles} \quad (2) \\ \{T,\epsilon\} & \textit{si } (1) \textit{ et } (2) \\ \emptyset & \textit{sinon} \end{cases}$$

Pour simplifier la notation dans la suite, on note $\{T,\epsilon\}$ par $T\epsilon$. L'idée de notre algorithme consiste à marquer chaque arc (i,j) dans $\mathcal{A}(\mathcal{T})$ par des labels $\mu_{i\to j}$ exprimant l'ensemble des informations contenues dans $\mathcal{V}_{\text{-}j}(i)$.

DÉFINITION 4 (Label-message μ). — *Pour $(i,j) \in \mathcal{A}(\mathcal{T})$, le label-message envoyé de i à j est la fonction $\mu : \mathcal{A}(\mathcal{T}) \mapsto 2^{\{\epsilon,T\}}$ telle que $\mu_{i\to j} := \bigcup_{\substack{k'\in\mathcal{V}_{\text{-}j}(i) \\ \{k\}=\mathrm{Adj}_j(k')}} \lambda_{k'\to k}$.*

Pour illustrer la discussion précédente, imaginons une première mise à jour incrémentale du DAG initial et, par conséquent, le JT initial dans la figure 5a. Il s'agit, par exemple, de la suppression de k_4, l'insertion d'une observation dans r et une nouvelle *cible* dans k_1. La figure 5b montre l'émission des μ-messages au sein du JT $\mathcal{T}$ après cette mise à jour, où les ellipses pointillées et en traits discontinus représentent respectivement les cliques contenant des observations et celles contenant des *cibles*. On peut facilement voir que, par exemple, $\mu_{i\to j} = \epsilon$, $\mu_{j\to i} = T$ et $\mu_{j\to k_2} = T\epsilon$. La proposition suivante permet de construire récursivement les messages μ :

PROPOSITION 5 (Construction de μ). — *Soit $(i,j) \in \mathcal{A}(\mathcal{T})$, alors on a : $\mu_{i\to j} = \lambda_{i\to j} \cup \bigcup_{k\in\mathrm{Adj}_{\text{-}j}(i)} \mu_{k\to i}$.*

---

6. Par abus, nous disons qu'une clique reçoit une observation quand au moins une de ses variables aléatoires a été observée. Étant donné que le comportement de l'inférence est identique au regard de ces deux transformations (observations et changements de structure), on va les identifier et appeler cela des observations tout court dans la suite.

Toutes les démonstrations des propositions et théorèmes se trouvent en annexe.

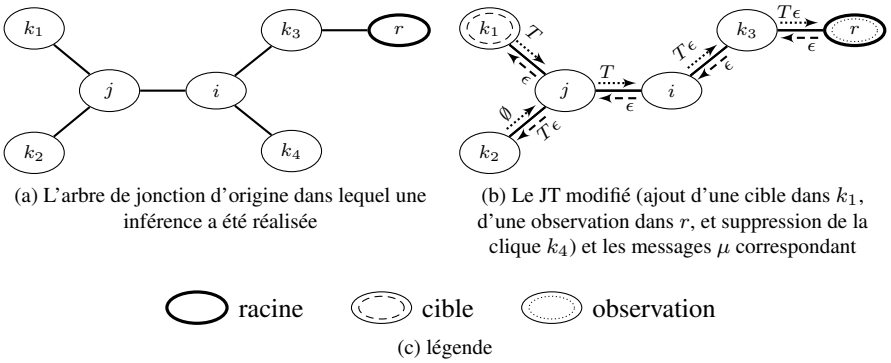
Une fois tous les nœuds du JT sont labellisés, chaque label-message est associé à un arc. Donc, on peut conclure que l'envoie des message μ peut se faire en temps linéaire en la taille du JT.

Rappelons que $\psi_{i\to j}$ dénote le message échangé entre les cliques i et j pendant un calcul d'inférence, à ne pas confondre avec le label-message $\mu_{i\to j}$ de la Définition 4.

4.3. Détermination de(s) la racine(s) optimale(s)

En général, le nombre de calculs effectués par un algorithme par envoi de messages dans un JT ne dépend pas de la clique racine choisie pour la *collecte/distribution*, car tous les messages $\psi_{i\to j}$ sont calculés *sur toutes* les arêtes du JT, c'est-à-dire que $2\times|\mathcal{E}|$ messages sont systématiquement calculés. Dans notre cas, la situation est différente. En effet, notre algorithme calcule et envoie *seulement* les messages $\psi_{i\to j}$ qui sont nécessaires pour le calcul des distributions *a posteriori* des *cibles*. Sur certaines arêtes, IJTI ne calculera donc pas certains $\psi_{i\to j}$ parce qu'ils ne sont pas pertinents pour le calcul de ces distributions. En conséquence, dans IJTI, le nombre de calculs effectués est sensible à la sélection de la racine : par exemple, dans le JT de la figure 5a, si la clique i est observée et j est la seule *cible*, seulement le message $\psi_{i\to j}$ de i vers j est nécessaire, ce qui est précisément envoyé si la clique i est choisie comme racine (ici, seulement une distribution est nécessaire). Par contre, si la clique k_4 est choisie comme racine, le message $\psi_{i\to k_4}$ doit être envoyé pendant la *collecte* et les messages $\psi_{k_4\to i}$ et $\psi_{i\to j}$ doivent être envoyés pendant la *distribution*. On voit clairement que cela n'est pas optimal.

Pour déterminer les racines optimales, définissons $\delta_{i\to j}(r)$ comme l'indicateur exprimant si le message $\psi_{i\to j}$ doit être recalculé (dans ce cas, $\delta_{i\to j}(r) = 1$) ou non ($\delta_{i\to j}(r) = 0$) lorsque r est choisi comme racine. Dans IJTI, nous cherchons donc à minimiser $\delta(r) = \sum_{(k',k)\in\mathcal{A}(\mathcal{T})} \delta_{k'\to k}(r)$, qui correspond au nombre total de mes-

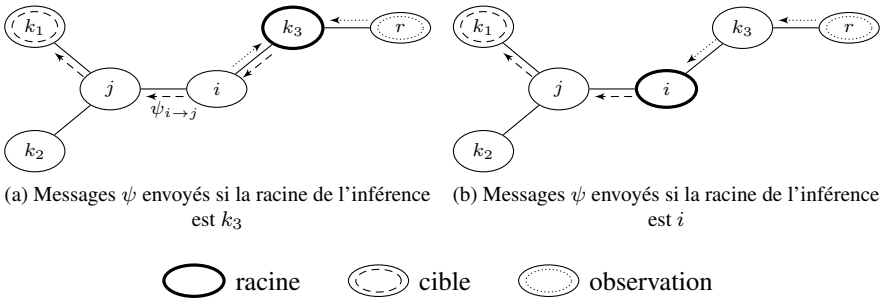


(a) L'arbre de jonction d'origine dans lequel une inférence a été réalisée

(b) Le JT modifié (ajout d'une cible dans k_1, d'une observation dans r, et suppression de la clique k_4) et les messages μ correspondant

(c) légende

Figure 5. Messages μ dans un JT $\mathcal{T}$



(a) Messages ψ envoyés si la racine de l'inférence est k_3

(b) Messages ψ envoyés si la racine de l'inférence est i

(c) légende

Figure 6. Les messages ψ dont le recalcul est nécessaire en fonction de la racine choisie. Les messages de collecte et de distribution sont respectivement indiqués en pointillés et en traits discontinus. Le sens des flèches indique la direction d'envoi

sages recalculés et envoyés. En se fondant sur la discussion de la section précédente, on peut écrire :

$$\delta_{i\to j}(r) = \begin{cases} 1 \text{ si } (\epsilon \in \mu_{i\to j} \text{ et } \{j\} = \mathrm{Adj}_r(i)) \text{ ou} \\ \qquad (\epsilon \in \mu_{i\to j} \text{ et } \{i\} = \mathrm{Adj}_r(j) \text{ et } T \in \mu_{j\to i}) \\ 0 \text{ sinon} \end{cases} \tag{1}$$

La première ligne de l'équation (1) concerne les messages de *collecte* ($\{j\} = \mathrm{Adj}_r(i)$). Elle stipule qu'un message $\psi_{i\to j}$ doit être recalculé seulement s'il est actuellement dans un « état invalide » ou si une nouvelle évidence ou des changements de structure affectent $\mathcal{V}_{\text{-}j}(i)$ ($\epsilon \in \mu_{i\to j}$). Quand ce n'est pas le cas, ce message est à jour et n'a pas besoin d'être recalculé. La deuxième ligne de l'équation (1) concerne les messages de *distribution* ($\{i\} = \mathrm{Adj}_r(j)$)). Elle stipule que $\psi_{i\to j}$ doit être recalculé seulement s'il existe une cible plus loin vers les feuilles du JT ($T \in \mu_{j\to i}$) et si des évidences ont été reçues dans $\mathcal{V}_{\text{-}j}(i)$ ou si certains messages venant de $\mathcal{V}_{\text{-}j}(i)$ ont été mis à jour ($\epsilon \in \mu_{i\to j}$). L'équation (1) peut être alors réécrite d'une façon plus compacte comme suit :

$$\delta_{i\to j}(r) = \begin{cases} 1 \text{ si } \epsilon \in \mu_{i\to j} \text{ et } (\{j\} = \mathrm{Adj}_r(i) \\ \qquad \text{ou } (\{i\} = \mathrm{Adj}_r(j) \text{ et } T \in \mu_{j\to i})) \\ 0 \text{ sinon} \end{cases} \tag{2}$$

Dans les figures 6a et 6b, $\delta(k_3) = 5$ et $\delta(i) = 4$. Dans ce cas, il vaut mieux choisir i comme racine plutôt que k_3, puisque cela nous épargne le calcul d'un message non nécessaire. Le théorème suivant montre l'existence des racines optimales et les caractérise.

THÉORÈME 6 (Racines optimales). — *Supposons que l'on a effectué le calcul des messages μ dans $\mathcal{T}$. Alors il existe $r \in \mathcal{V}(\mathcal{T})$ satisfaisant l'une des propriétés exhaustives et mutuellement exclusives suivantes :*

1. $(\mathcal{V}(\mathcal{T}), \mathcal{E}(\mathcal{T})) = (\{r\}, \emptyset)$
2. $\exists r' \in \mathcal{V}(\mathcal{T}) : \mu_{r'\to r} = \mu_{r\to r'} = T\epsilon$

3. $\forall k \in \mathrm{Adj}(r) : \mu_{k \to r} \in \{T, \epsilon, \emptyset\}$

De plus, $r \in \mathrm{Argmin}_{k \in \mathcal{V}(\mathcal{T})} \delta(k)$*, c'est-à-dire que* r *est un nœud optimal par rapport aux calculs de l'inférence.*

Remarquons que pour trouver une racine optimal, il suffit de tester les labels des arcs jusqu'à vérifier l'une des propriétés du théorème précédent. En conséquence, trouver une racine optimale se fait en temps linéaire en la taille du JT dans le pire cas.

4.4. Une nouvelle inférence incrémentale

Dans ce paragraphe, nous proposons un nouvel algorithme conçu pour répondre au problème de l'incrémentalité dans l'inférence.

Algorithme 1 : IJTI

input : $\mathcal{T}$ modifié, Q cliques des *cibles*
output : distributions *a posteriori* des *cibles*
// initialiser le nombre
// des voisins visités
// durant la collecte
pour $i \in \mathcal{V}(\mathcal{T})$ **faire**
  $i.nbVN \leftarrow 0$
Calculer les μ-labels de $\mathcal{T}$
Trouver r par le théorème 6
$\mathbf{L} \leftarrow$ l'ensemble des feuilles de $\mathcal{T}$
// phase de collecte
pour chaque *clique* $i \in \mathbf{L}$ **faire**
  $p \leftarrow \mathrm{Adj}_r(i)$
  si $\delta_{i \to p}(r) = 1$ **alors**
    Calculer $\psi_{i \to p}$
  $p.nbVN \leftarrow p.nbVN + 1$
  si $p \neq r$ **et** $p.nbVN = |\mathrm{Adj}(p)| - 1$ **alors** $\mathbf{L} \leftarrow \mathbf{L} \cup \{p\}$
  $\mathbf{L} \leftarrow \mathbf{L} \setminus \{i\}$
// phase de distribution
$\mathbf{L} \leftarrow \{r\}$
pour chaque *clique* $i \in \mathbf{L}$ **faire**
  pour chaque $j \in \mathrm{Adj}(i) \setminus \mathrm{Adj}_r(i)$ **faire**
    si $\delta_{i \to j}(r) = 1$ **alors**
      Calculer $\psi_{i \to j}$
      $\mathbf{L} \leftarrow \mathbf{L} \cup \{j\}$
pour chaque *clique* $t \in Q$ **faire**
  Calculer les distributions *a posteriori* des *cibles* dans la clique t
retourner les distributions *a posteriori*

On suppose qu'une première inférence par envoi de messages a été déjà effectuée dans $\mathcal{T}$ en utilisant, par exemple, un algorithme de *collecte-distribution* dans une architecture de type *Lazy Propagation*. Des changements incrémentaux apparaissent par la suite et *IJTI* est appelé pour optimiser l'inférence. On rappelle que notre approche est dirigée par les *cibles*, ainsi, on recalcule seulement les messages de *collecte* invalides et on *distribue* les messages seulement vers les *cibles*. Ayant considéré ces hypothèses, l'algorithme proposé est décrit dans l'algorithme 1. Il s'agit d'un algo-

rithme révisé d'envoi de messages pour calculer $\psi_{i\to j}$ seulement lorsque $\delta_{i\to j}(r) = 1$ pour tout i, j dans le JT modifié $\mathcal{T}$. Dans la ligne 5, une clique feuille i est telle que $|\mathrm{Adj}(i)| = 1$. Nous signalons que le calcul des messages se fait de manière similaire aux algorithmes classiques basés sur l'utilisation du JT. La proposition suivante garantie la justesse de l'algorithme IJTI :

PROPOSITION 7. — *l'algorithme IJTI est valide, c'est-à-dire que le calcul des seuls messages $\psi_{i\to j}$ pour lesquels $\delta_{i\to j}(r) = 1$, pour tout $(i, j) \in \mathcal{A}(\mathcal{T})$, permet de calculer correctement les distributions* a posteriori *des variables* cibles.

5. Résultats expérimentaux

Dans cette section, nous mettons en évidence l'efficacité de notre algorithme via la comparaison du gain obtenu avec son utilisation à la place d'un autre algorithme d'inférence de JT non-incrémental.

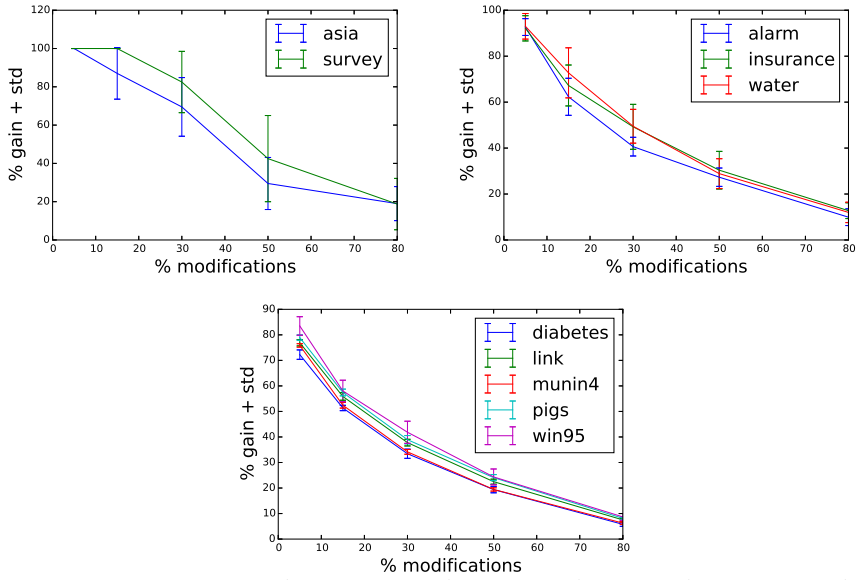


*Figure 7. Pourcentage de gain moyen d'*IJTI *pour des BN réels en fonction du pourcentage de modifications apportées à l'arbre de jonction (évidences, targets, modifications de structure.) Les courbes représentent les moyennes de gains sur 20 requêtes tandis que les barres verticales représentent les écarts types*

Ce gain vaut $1 - \delta(r)/(2|\mathcal{E}(\mathcal{T})|)$, c'est-à-dire le pourcentage des messages non nécessaires que IJTI évite de calculer par rapport aux algorithmes d'inférence classiques qui envoient des messages dans les deux directions sur toutes les arêtes. Il est à noter que nous effectuons des comparaisons en nombre de messages. Le gain "réel" nécessite des comparaisons en temps de calculs, mais cette métrique est toutefois plus dispersée et donc difficile à interpréter car elle dépend fortement des domaines des

variables ainsi que de la taille des cliques (ces deux paramètres variant fortement d'un réseau bayésien à l'autre dans nos tests). Notons que, dans une inférence, les temps de calcul correspondent essentiellement aux opérations de combinaisons et de projections des tables de probabilité, le temps nécessaire à la construction de l'arbre de jonction et de la détermination des messages μ et δ étant marginal par rapport à ces opérations algébriques. C'est la raison pour laquelle la notion de gain que nous utilisons dans ces expériences est pertinente.

Pour nos expérimentations, nous avons effectué des tests en utilisant la librairie aGrUM [7] sur neuf BN réels de complexités différentes ainsi que sur des BN aléatoirement générés. Les BN réels, `asia`, `survey`, `alarm`, `insurance`, `water`, `diabetes`, `link`, `munin4`, `pigs` et `win95` sont des BN classiques provenant du *repository* classique de BNlearn : `http://www.bnlearn.com/bnrepository`. Leurs descriptions précises (nombre de nœuds, arcs, *etc.*) sont indiqués sur cette URL. Les BN ont été générés aléatoirement en utilisant les outils de la librairie aGrUM et contiennent $\mathrm{nbNodes}$ variables aléatoires de telle sorte que $6 \leq \mathrm{nbNodes} \leq 900$ (voir la figure 8) et, pour chaque valeur de $\mathrm{nbNodes}$, trois BN sont générés avec nbArcs arcs, nbArcs est choisi aléatoirement dans l'intervalle $[\mathrm{nbNodes}-1, 4/3*\mathrm{nbNodes}-1]$. Les tables de probabilité ont été remplies aléatoirement puisque cela n'a aucun impact sur les résultats des expériences.

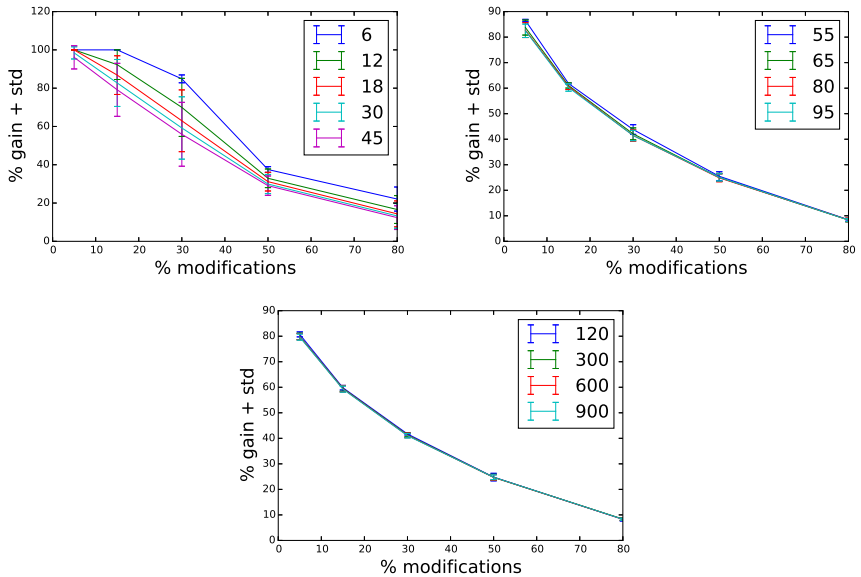


Figure 8. Pourcentage de gain d'IJTI pour des BN artificiels (de 6 à 900 nœuds)

Pour simuler l'incrémentalité et induire des messages invalides dans $\mathcal{T}$, nous avons choisi aléatoirement, pour chaque inférence, un ensemble de *cibles* et d'observations.

7. `http://agrum.lip6.fr`

Chaque nœud choisi est considéré donc comme une modification du BN. Les figures 7 et 8 montrent les gains moyens résultants et leurs écarts type (les barres d'erreurs [8]) sur 20 requêtes d'inférence incrémentale. Notons que le comportement de l'algorithme est le même pour les BN réels et artificiels. Comme l'on pouvait s'y attendre, plus les modifications sont petites, plus le gain est grand. Notons également que le gain est très sensible à la taille du BN.

6. Conclusion

Dans cet article, nous avons proposé d'exploiter les modèles probabilistes relationnels (PRM) afin de modéliser les incertitudes dans les systèmes de gestion de règles métier orientés objets (OO-BRMS). Les PRM sont en effet particulièrement bien adaptés pour cette tâche dans la mesure où leur modèle orienté objet permet de « coller » au maximum aux objets des OO-BRMS. Malheureusement, les méthodes d'inférence dans les PRM ne sont pas optimales dans la mesure où elles n'exploitent pas les redondances entre calculs d'une inférence à la suivante. Afin de pallier ce problème et d'accélerer significativement les inférences, et donc les temps de réponse des OO-BRMS, nous avons introduit *IJTI*, un nouvel algorithme d'inférence incrémentale par JT conçu pour les systèmes multi-cibles dynamiques. En supposant qu'une première inférence complète a été effectuée, il extrait pour chaque inférence ultérieure une racine optimale et minimise le nombre des messages à calculer pour cette inférence en conséquence. La justesse mathématique de ces deux optimisations est prouvée et la partie expérimentale souligne des économies importantes de notre approche comparée aux méthodes classiques. Concernant de futurs travaux, nous visons l'amélioration de notre algorithme en considérant notamment des techniques de caches pour la détermination des racines. Nous envisageons également d'exploiter dans IJTI des particularités des modèles probabilistes relationnels, notamment de conjuguer IJTI avec de l'inférence structurée (Wuillemin, Torti, 2012). De plus, on pourrait aussi penser à construire le BN *groundé* de manière incrémentale au fur et à mesure de la mise à jour de la mémoire du travail. Cela devrait permettre d'accélérer encore l'inférence dans les PRM et donc dans les OO-BRMS.

Remerciements

Ce travail a été partiellement financé par IBM France Lab, bourse CIFRE ANRT #2014/421.

8. Dans la figure 7.a, le fait d'avoir un petit nombre de nœuds et d'arcs dans les BN implique l'absence de toute modification pour les pourcentages de modifications inférieurs à 10%, d'où l'absence des barres d'erreurs.

Appendice : Démonstrations

Démonstration de la proposition 5 : Notons que $\mathcal{V}_{\text{-}j}(i) = \{i\} \cup \bigcup_{k \in \mathrm{Adj}_{\text{-}j}(i)} \mathcal{V}_{\text{-}i}(k)$ et pour $k \in \mathrm{Adj}_{\text{-}j}(i)$, $l' \in \mathcal{V}_{\text{-}i}(k)$, on a $\mathrm{Adj}_j(l') = \mathrm{Adj}_i(l')$. En utilisant la définition 4, on peut ainsi réécrire $\mu_{i\to j}$ comme suit :

$$\mu_{i\to j} = \bigcup_{\substack{l' \in \mathcal{V}_j(i) \\ \{l\} = \mathrm{Adj}_j(l')}} \lambda_{l'\to l} = \lambda_{i\to j} \cup \bigcup_{k \in \mathrm{Adj}_{\text{-}j}(i)} \overbrace{\bigcup_{\substack{l' \in \mathcal{V}_i(k) \\ \{l\} = \mathrm{Adj}_j(l')}} \lambda_{l'\to l}}^{\mu_{k\to i}} = \lambda_{i\to j} \cup \bigcup_{k \in \mathrm{Adj}_{\text{-}j}(i)} \mu_{k\to i}$$

■

Démonstration de l'exclusivité mutuelle des propriétés du théorème 6 : si la propriété 1) est vérifiée, alors $\mathcal{T}$ ne contient aucune arête, et par conséquent, les propriétés 2) et 3) ne peuvent pas être satisfaites.

Supposons maintenant qu'il existe r_1, r_1' tels que $\mu_{r_1'\to r_1} = \mu_{r_1\to r_1'} = T\epsilon$ (propriété 2). soit r_2 une clique quelconque de $\mathcal{V}(\mathcal{T})$. Sans perte de généralité, supposons que r_1 appartienne à la chaîne $i_1 = r_2, i_2, \ldots, i_p = r_1'$ entre r_2 et r_1'. Alors, en appliquant la proposition 5, $\mu_{i_2\to r_2} \supseteq \mu_{i_3\to i_2} \supseteq \cdots \supseteq \mu_{r_1'\to r_1} = T\epsilon$. Par conséquent, les propriétés 2) et 3) sont insatisfiables simultanément. ■

Démonstration de l'existence de r du théorème 6 : si $\mathcal{A}(\mathcal{T}) = \emptyset$, alors la propriété 1) est vraie et r est l'unique nœud de $\mathcal{T}$. Dans le reste de la démonstration, supposons que $\mathcal{A}(\mathcal{T}) \neq \emptyset$. S'il existe une arête $(i,j) \in \mathcal{E}(\mathcal{T})$ telle que $\mu_{i\to j} = \mu_{j\to i} = T\epsilon$, alors $r = i$ vérifie la propriété b). Sinon, les propriétés 1) et 2) ne sont plus vraies. Supposons que la propriété 3) n'est pas vérifiée non plus. Alors pour toutes les arêtes (i,j), entre $\mu_{i\to j}$ et $\mu_{j\to i}$, un seul de ces messages est égal à $T\epsilon$ et l'autre appartient à $\{\emptyset, \epsilon, T\}$.

Soit (i_0, j_0) telle que $\mu_{i_0\to j_0} = T\epsilon$ et $\mu_{j_0\to i_0} \neq T\epsilon$. Alors, si $|\mathrm{Adj}(i_0)| = 1$, la clique i_0 satisfait la propriété 3), ce qui est une contradiction. Comme nous avons aussi supposé que la propriété 2) n'est pas vérifiée, il existe $i_1 \in \mathrm{Adj}(i_0)$ tel que $\mu_{i_1\to i_0} = T\epsilon$ et $\mu_{i_0\to i_1} \neq T\epsilon$. Le même raisonnement est toujours vrai pour i_1. Si i_1 est une feuille, on obtient une contradiction avec la propriété 3). Sinon, i_1 possède un voisin i_2 tel que $\mu_{i_2\to i_1} = T\epsilon$ et $\mu_{i_1\to i_2} \neq T\epsilon$. Par récurrence, on crée une chaîne $i_1, \ldots, i_n$ de taille maximale. Cette chaîne est forcément finie puisque $\mathcal{T}$ est lui-même un arbre fini, ainsi la clique i_n est une feuille qui, de plus, satisfait la propriété, ceci est encore une fois une contradiction. En conséquence, lorsque les propriétés 1) et 2) ne sont pas satisfaites, la propriété 3) est forcément vérifiée. ■

Dorénavant, on peut prouver séparément l'optimalité de chaque propriété du théorème 6, puisque ces propriétés sont mutuellement exclusives :

Démonstration de l'optimalité de la propriété 1) du théorème 6 : r est la seule racine à prendre. ■

LEMME 8. — *Soient* $i, j \in \mathcal{V}(\mathcal{T})$ *tels que* $\epsilon \in \mu_{j\to i}$ *et* $\mu_{i\to j} = \emptyset$, *alors* $\forall l \in \mathcal{V}_{\text{-}j}(i)$: $\delta(l) = \delta(j) + \text{len}(l-j)$.

Démonstration. Notons que lorsque $\epsilon \notin \mu_{j\to i}$, $\mathcal{T}$ est à jour dans l'inférence actuelle, on n'aura donc pas besoin d'effectuer des calculs. La preuve est établie par récurrence sur $n = \text{len}(l-j)$. Pour $n = 1$, on a $l = i$, donc par l'équation (2) et le fait que $\epsilon \in \mu_{j\to i}$ et $i \in \text{Adj}_i(j)$, on obtient $\delta_{j\to i}(i) = 1$. Par conséquent, $\delta(i) = \sum_{(k',k)\in\mathcal{A}(\mathcal{T})\setminus\{(j,i)\}} \delta_{k'\to k}(i) + 1$. Or, puisque $T \notin \mu_{i\to j}$, on a $\delta_{j\to i}(j) = 0$; donc $\delta(j) = \sum_{(k',k)\in\mathcal{A}(\mathcal{T}))\setminus\{(j,i)\}} \delta_{k'\to k}(j)$. Puisque $\epsilon \notin \mu_{i\to j}$, $\delta_{i\to j}(i) = \delta_{i\to j}(j) = 0$. Pour $(k', k) \neq (i, j), (j, i)$, on a $\text{Adj}_i(k) = \text{Adj}_j(k)$ et $\text{Adj}_i(k') = \text{Adj}_j(k')$. Dans ce cas, il s'ensuit que $\delta_{k'\to k}(i) = \delta_{k'\to k}(j)$. On en conclue donc que $\delta(i) = \delta(j) + 1$.

Supposons maintenant que cette propriété est vraie jusqu'à l'ordre $n - 1 > 1$, montrons qu'elle le restera pour n. soit l tel que $\text{len}(l-j) = n - 1$ et $\{p\} = \text{Adj}_i(l)$. Alors $\delta(l) = 1 + \sum_{(k',k)\in\mathcal{A}(\mathcal{T})\setminus\{(p,l)\}} \delta_{k'\to k}(l)$ car $\delta_{p\to l}(l) = 1$ (puisque $\epsilon \in \mu_{p\to l}$ et $\{l\} = \text{Adj}_l(p)$). Étant donné que $T \notin \mu_{l\to p}$, on obtient $\delta_{p\to l}(p) = 0$, ce qui implique que $\delta(p) = \sum_{(k',k)\in\mathcal{A}(\mathcal{T})\setminus\{(p,l)\}} \delta_{k'\to k}(p)$. Maintenant, en utilisant le même raisonnement que pour le cas $n = 1$ et en remarquant que $\delta_{l\to p}(p) = \delta_{l\to p}(l) = 0$ car $\epsilon \notin \mu_{l\to p}$, on conclut que $\delta(l) = 1 + \sum_{(k',k)\in\mathcal{A}(\mathcal{T})\setminus\{(p,l)\}} \delta_{k'\to k}(l) = 1 + \sum_{(k',k)\in\mathcal{A}(\mathcal{T})\setminus\{(p,l)\}} \delta_{k'\to k}(p) = 1 + \delta(p)$. L'hypothèse du récurrence appliquée à l, où $\text{len}(l-j) = n-1$, assure que : $\delta(l) = 1+\delta(p) = 1+n-1+\delta(j) = \delta(j)+n$. ■

LEMME 9. — *Soit* $\mathcal{V}_1 = \{r \in \mathcal{V}(\mathcal{T}) : \exists k \in \text{Adj}(r), \mu_{r\to k} = \mu_{k\to r} = T\epsilon\}$, *alors pour tout* $r, r' \in \mathcal{V}_1$ *on a* $\delta(r) = \delta(r')$.

Démonstration. Supposons que $|\mathcal{V}_1| > 1$. D'après la proposition 5, les nœuds dans $\mathcal{V}_1$ forment un sous graphe connexe. Soient $r, r' \in \mathcal{V}_1$ tels que $(r, r') \in \mathcal{E}(\mathcal{T})$. On note $\mathcal{A}_{r,r'}(\mathcal{T}) = \mathcal{A}(\mathcal{T}) \setminus \{(r, r'), (r', r)\}$. Soit $(k', k) \in \mathcal{A}_{r,r'}(\mathcal{T})$. Si $k' \notin \{r, r'\}$, alors soit $k = r$, $k = r'$ soit $k \notin \{r, r'\}$ et dans tous les cas on a : $\text{Adj}_r(k') = \text{Adj}_{r'}(k')$, ainsi $\delta_{k'\to k}(r) = \delta_{k'\to k}(r')$. Sinon, soit $k' = r'$ alors $k \neq r$ et on a aussi [9] $\text{Adj}_r(k) = \text{Adj}_{r'}(k)$ et encore une fois, $\delta_{k'\to k}(r) = \delta_{k'\to k}(r')$. Par conséquent, $\sum_{(k',k)\in\mathcal{A}_{r,r'}(\mathcal{T})} \delta_{k'\to k}(r) = \sum_{(k',k)\in\mathcal{A}_{r,r'}(\mathcal{T})} \delta_{k'\to k}(r')$. En utilisant l'équation (2), on obtient $\delta_{r\to r'}(r) + \delta_{r'\to r}(r) = \delta_{r\to r'}(r') + \delta_{r'\to r}(r') = 2$. On en conclut que $\delta(r) - \sum_{(k',k)\in\mathcal{A}_{r,r'}(\mathcal{T})} \delta_{k'\to k}(r) = \delta(r') - \sum_{(k',k)\in\mathcal{A}_{r,r'}(\mathcal{T})} \delta_{k'\to k}(r')$. Ainsi $\delta(r) = \delta(r')$. ■

Démonstration de l'optimalité de la propriété 2 du théorème 6 : Avec les notations de la propriété 2), il suffit de prouver que pour tout i n'appartenant pas à $\mathcal{V}_1, \delta(r) \leq \delta(i)$ [10]. Sans perte de généralité, supposons que $i \in \mathcal{V}_{\text{-}r'}(r)$. Soit $(k, k') \in \mathcal{A}(i-r)$, où $\mathcal{A}(i-r)$ est l'ensemble des arcs induits de $i-r$. Alors on a soit $\{k'\} =$

9. si $k' = r$ alors $k \neq r'$ et l'égalité est vérifiée.

10. Tous les nœuds sont équivalents en termes de calcul si $\forall i \in \mathcal{V}(\mathcal{T}), i \in \mathcal{V}_1$, puisque $\mathcal{V}(\mathcal{T}) = \mathcal{V}_1$

$\mathrm{Adj}_r(k)$ soit $\{k\} = \mathrm{Adj}_r(k')$. Supposons par exemple que $\{k'\} = \mathrm{Adj}_r(k)\,, k \neq r$, le deuxième cas est traité de la même façon. Alors $\mu_{k'\to k} = T\epsilon$, et par application de l'équation (2), on récapitule les résultats dans le tableau suivant :

$\mu_{k\to k'}$	$\delta_{k\to k'}(i) + \delta_{k'\to k}(i)$	$\delta_{k\to k'}(r) + \delta_{k'\to k}(r)$
$\emptyset$	1	0
T	1	1
ϵ	2	1

On conclut que $\sum_{(k',k)\in\mathcal{A}(i-r)} \delta_{k'\to k}(r) \leq \sum_{(k',k)\in\mathcal{A}(i-r)} \delta_{k'\to k}(i).(1)$
Maintenant pour $(k,k') \notin \mathcal{A}(i-r)$, il est facile de voir que $\delta_{k\to k'}(i) = \delta_{k\to k'}(r)$ et, par conséquent, $\sum_{(k,k')\in\mathcal{A}(\mathcal{T})\setminus\mathcal{A}(i-r)} \delta_{k'\to k}(r) = \sum_{(k,k')\in\mathcal{A}(\mathcal{T})\setminus\mathcal{A}(i-r)} \delta_{k'\to k}(i).$ (2). En comparant (1) et (2), on obtient $\delta(r) \leq \delta(i)$ pour $i \notin \mathcal{V}_1$. Jusqu'ici, nous avons obtenu, par le lemme 9, que pour tout i dans $\mathcal{V}_1$, $\delta(r) = \delta(i)$ et pour tout i n'appartenant pas à $\mathcal{V}_1$, on a $\delta(r) \leq \delta(i)$. On en déduit donc que $r \in \mathrm{Argmin}_{i\in\mathcal{V}(\mathcal{T})}$ $\delta(i)$. ■

Démonstration de l'optimalité de la propriété 3) du théorème 6 : soit $i \in \mathcal{V}(\mathcal{T})$ tel que $i \neq r$.
premier cas : $\mu_{\mathrm{Adj}_i(r)\to r} = \emptyset$. Supposons que $T, \epsilon \in \mathcal{V}_{\text{-}i}(r)$, car sinon il n'y a aucun besoin d'effectuer un quelconque calcul d'inférence, puisqu'il s'agit d'une absence de requête ou de modifications dans $\mathcal{T}$; donc par le lemme 8, on a $\delta(i) = \delta(r) + \mathrm{len}(i–r)$ car $i \in \mathcal{V}_{\text{-}r}(\mathrm{Adj}_i(r))$. Par conséquent, $\delta(r) < \delta(i)$.
deuxième cas : Nous omettons le cas $\mu_{\mathrm{Adj}_i(r)\to r} \in \{T, \epsilon\}$, mais le lecteur pourra utiliser la même méthode que dans la preuve de la propriété 2) et le fait que pour tout $k, k' \in i-r$ tels que $\{k'\} = \mathrm{Adj}_r(k)$: $\mu_{k\to k'} = \mu_{i\to \mathrm{Adj}_r(i)}$ et examiner $\delta_{k'\to k}(r)$ et $\delta_{k'\to k}(i)$. ■

Démonstration de la proposition 7 : Étant donné une racine r, $\delta_{i\to j}(r)$ correspond, par construction, au fait que $\psi_{i\to j}$ est nécessaire durant l'inférence actuelle et a été invalidé durant la précédente. En conséquence, l'inférence actuelle a besoin uniquement de recalculer un tel message pour tout i, j dans $\mathcal{V}(\mathcal{T})$. ■

Bibliographie

Allen D., Darwiche A. (2003). New advances in inference by recursive conditioning. In *Proceedings of the conference on uncertainty in artificial intelligence (UAI)*, p. 2–10.

Bacchus F., Dalmao S., Toniann Pitassi T. (2003). DPLL with caching: A new algorithm for #SAT and Bayesian inference. In *Electronic colloquium on computational complexity (ecc).*

Beinlich I. A., Suermondt H. J., Chavez R. M., Cooper G. F. (1989). The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the 2nd european conference on artificial intelligence in medicine*, p. 247–256. Springer-Verlag.

Berstel-Da Silva B. (2014). *Verification of business rules programs*. Springer.

Buchanan B. G., Shortliffe E. H. (1984). *Rule based expert systems: The Mycin experiments of the Stanford heuristic programming project*. Addison-Wesley.

Chavira M., Darwiche A. (2008). On probabilistic inference by weighted model counting. *Artificial Intelligence Journal*, vol. 172, n° 6-7, p. 772–799.

D'Ambrosio B. (1993). Incremental probabilistic inference. In *Proceedings of the 9th international conference on uncertainty in artificial intelligence (UAI)*, p. 301–308.

Darwiche A. (1998). Dynamic Join Trees. In *Proceedings of the 14th conference on uncertainty in artificial intelligence (UAI)*, p. 97–104.

Darwiche A. (2009). *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press.

Dean T., Kanazawa K. (1989). A model for reasoning about persistence and causation. *Computational Intelligence*, vol. 5, p. 142–150.

Dubuisson S., Gonzales C., Nguyen X.-S. (2012). DBN-based combinatorial resampling for articulated object tracking. In *Proceedings of conference on uncertainty in artificial intelligence (UAI)*, p. 237–246.

Elkan C. (1993). The paradoxical success of fuzzy logic. In *Proceedings of the 11th national conference on artificial intelligence (AAAI)*, p. 698–703.

Flores M. J., Gámez J. A., Olesen K. G. (2003). Incremental Compilation of Bayesian Networks. In *Proceedings of the 19th conference on uncertainty in artificial intelligence (UAI)*, p. 233–240.

Forgy C. (1982). Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. *Artificial Intelligence*, vol. 19, p. 17–37.

Gonzales C., Wuillemin P.-H. (2011). PRM inference using Jaffray and Faÿ's local conditioning. *Theory and Decision*, vol. 71, n° 1, p. 33–62.

Graham I. (2007). *Business rules management and service oriented architecture: A pattern language*. Wiley.

Hart P. E., Duda R. O., Einaudi M. T. (1977). PROSPECTOR—a computer-based consultation system for mineral exploration. *Journal of the International Association for Mathematical Geology*, vol. 10, p. 589–610.

Heckerman D. E., Shortliffe E. H. (1992). From certainty factors to belief networks. *Artificial Intelligence in Medicine*, vol. 4, p. 35–52.

Jensen F., Lauritzen S., Olesen K. (1990). Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, vol. 4, p. 269–282.

Koller D., Friedman N. (2009). *Probabilistic graphical models: Principles and techniques*. MIT Press.

Koller D., Pfeffer A. (1998). Probabilistic frame-based systems. In *Proceedings of the 15th national conference on artificial intelligence (AAAI)*, p. 580–587.

Lauritzen S., Spiegelhalter D. J. (1988). Local computations with probabilities on graphical structures and their applications to expert systems. *Journal of the Royal Statistical Society*, vol. 50, p. 157–224.

Li W., Beek P. van, Poupart P. (2006). Performing Incremental Bayesian Inference by Dynamic Model Counting. In *Proceedings of the national conference on artificial intelligence (AAAI)*, p. 1173-1179.

Lin Y., Druzdzel M. J. (1998). Relevance-Based Sequential Evidence Processing in Bayesian Networks. In *Proceedings of the 11th international florida artificial intelligence research society conference (FLAIRS)*, p. 446–450.

Madsen A. L., Jensen F. V. (1999). Lazy propagation: A junction tree inference algorithm based on lazy evaluation. *Artificial Intelligence*, vol. 113, p. 203–245.

Murphy K. P. (2002). *Dynamic Bayesian networks: Representation, inference and learning*. Thèse de doctorat non publiée, University of California at Berkeley.

Naim P., Wuillemin P.-H., Leray P., Pourret O., Becker A. (2007). *Réseaux bayésiens*. Eyrolles.

Ng K.-C., Abramson B. (1990). Uncertainty management in expert systems. *IEEE Expert: Intelligent Systems and Their Applications*, vol. 5, p. 29–48.

Pearl J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann.

Pfeffer A. J. (2000). *Probabilistic reasoning for complex systems*. Thèse de doctorat non publiée, Stanford University.

Robinson J. W., Hartemink A. J. (2009). Non-stationary dynamic Bayesian networks. In D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), *Advances in neural information processing systems 21*, p. 1369–1376. Curran Associates, Inc.

Sang T., Bearne P., Kautz H. (2005). Performing Bayesian Inference by Weighted Model Counting. In *Proceedings of the 20th national conference on artificial intelligence (AAAI)*, p. 475–481.

Shachter R. D. (1998). Bayes-ball: Rational Pastime (for Determining Irrelevance and Requisite Information in Belief Networks and Influence Diagrams). In *Proceedings of the fourteenth conference on uncertainty in artificial intelligence (UAI)*, p. 480–487.

Shenoy P., Shafer G. (2008). Axioms for Probability and Belief-Function Propagation. In R. R. Yager, L. Liu (Eds.), *Classic works of the dempster-shafer theory of belief functions*, p. 499–528. Springer Berlin Heidelberg.

Torti L. (2012). *Inférence probabiliste structurée dans les modèles graphiques probabilistes orientés-objet*. Thèse de doctorat non publiée, Pierre et Marie Curie University.

Torti L., Gonzales C., Wuillemin P.-H. (2013). Speeding-up structured probabilistic inference using pattern mining. *International Journal of Approximate Reasoning*, vol. 54, p. 900–918.

Torti L., Wuillemin P.-H., Gonzales C. (2010). Reinforcing the object-oriented aspect of probabilistic relational models. In *Proceedings of the 5th conference on probabilistic graphical models (PGM)*, p. 273–280.

Wuillemin P.-H., Torti L. (2012). Structured probabilistic inference. *International Journal of Approximate Reasoning*, vol. 53, p. 946 - 968.

Zadeh L. A. (1965). Fuzzy sets. *Information and Control*, vol. 8, p. 338–353.