1 Les modèles graphiques

1.1 Introduction

Les modèles graphiques portent de nombreux noms : réseaux de croyance, réseaux probabilistes, réseaux d'indépendance probabiliste ou encore réseaux bayésiens. Il s'agit d'un formalisme pour représenter de façon factorisée une distribution jointe de probabilités sur un ensemble de variables aléatoires. Ils ont révolutionné le développement des systèmes intelligents dans de nombreux domaines.

Ils sont le mariage entre la théorie des probabilités et la théorie des graphes. Ils apportent des outils naturels permettant de traiter deux grands problèmes couramment rencontrés en intelligence artificielle, en mathématiques appliquées ou en ingénierie : l'incertitude et la complexité. Ils jouent en particulier un rôle grandissant dans la conception et l'analyse d'algorithmes liés au raisonnement ou à l'apprentissage [Jordan, 1999], [Becker and Naïm, 1999], [Dawid, 1992].

A la base des modèles graphiques se trouve la notion fondamentale de la *modularité* : un système complexe est construit par la combinaison de parties simples !

La théorie des probabilités fournit le ciment permettant la combinaison de ces parties, tout en assurant que le modèle est et reste consistant. Elle fournit un moyen d'interfacer modèles et données. La théorie des graphes apporte d'une part une interface intuitive grâce à laquelle un humain peut modéliser un problème comportant des variables interagissant entre elles, d'autre part un moyen de structurer les données. Ceci mène alors vers une conception naturelle d'algorithmes génériques efficaces.

Beaucoup de systèmes probabilistes classiques issus de domaines tels que les statistiques, la théorie de l'information, la reconnaissance des formes ou encore la mécanique statistique, sont en fait des cas particuliers du formalisme plus général que constituent les modèles graphiques. Dans ce domaine, on peut citer les modèles de Markov cachés, les filtres de Kalman ou encore les modèles d'Ising [Jordan, 1999]. Ainsi, les modèles graphiques sont un moyen efficace de voir tous ces systèmes comme des instances d'un formalisme commun sous-jacent.

L'avantage immédiat réside dans le fait que les techniques développées pour certains domaines peuvent alors être aisément transférées à un autre domaine et être exploitées plus facilement. Les modèles graphiques fournissent ainsi un formalisme naturel pour la conception de nouveaux systèmes [Jordan, 1999].

Cette synthèse a pour but de clarifier le domaine et de présenter les derniers résultats au niveau de la théorie sur les réseaux bayésiens. Elle est nécessaire en particulier pour comprendre le fonctionnement de l'inférence exacte et des algorithmes associés.

Dans un premier temps, je vais présenter les définitions et les théorèmes de base sur les réseaux bayésiens. Ensuite je parlerai de la notion d'indépendance conditionnelle, qui est à la base du processus de représentation de la connaissance dans les réseaux bayésiens. La seconde partie portera sur l'inférence et en particulier sur l'algorithme JLO dit algorithme de l'arbre de jonction. Il s'agit d'un algorithme d'inférence exacte. Je terminerai sur une conclusion et sur quelques perspectives.

1.2 Les réseaux bayésiens

1.2.1 Introduction

Dans cette section, nous allons nous focaliser sur un modèle particulier de la famille des modèles graphiques : les réseaux bayésiens, qui utilisent des graphes dirigés acycliques.

Le rôle des graphes dans les modèles probabilistes et statistiques est triple :

1. fournir un moyen efficace d'exprimer des hypothèses,
2. donner une représentation économique des fonctions de probabilité jointe,
3. faciliter l'inférence à partir d'observations.

Soit un ensemble U de variables aléatoires suivant une loi de Bernouilli, $U = \{x_1, x_2, \ldots, x_n\}$. Pour être stockée, la probabilité jointe $P(U)$ de cet ensemble nécessitera un tableau comprenant 2^n entrées : une taille particulièrement grande, quelque soit le système utilisé. Par contre, si nous savons que certaines variables ne dépendent en fait que d'un certain nombre d'autres variables, alors nous pouvons faire une économie substantielle en mémoire et par conséquent en temps de traitement. De telles dépendances vont nous permettre de décomposer cette très large distribution en un ensemble de distributions locales beaucoup plus petites, chacune ne s'intéressant qu'à un petit nombre de variables. Les dépendances vont aussi nous permettre de relier ces petites distributions en un grand ensemble décrivant le problème que l'on veut modéliser. On pourra ainsi répondre de façon cohérente et efficace à diverses questions que l'on pourrait se poser sur cette distribution de probabilités. Dans un graphe, il est possible de représenter chaque variable du problème par un noeud et chaque dépendance entre les variables par un arc.

Les graphes dirigés et non dirigés sont abondamment utilisés pour faciliter une telle décomposition des connaissances. Les modèles à base de graphes non dirigés sont souvent appelés *champs de Markov* [Pearl, 1988] et ont servi initialement à représenter des relations temporelles (ou spatiales) symétriques [Isham, 1981, Cox and Wermuth, 1996, Lauritzen, 1996]. Les graphes dirigés acycliques sont utilisés pour représenter des relations temporelles ou causales, en particulier dans [Lauritzen, 1982], [Wermuth and Lauritzebn, 1983] et [Kiiveri et al., 1984]. Judea Pearl les a nommés *Réseaux Bayésiens* en 1985 pour mettre en évidence trois aspects :

1. la nature subjective des informations,
2. l'utilisation de la règle de Bayes comme principe de base pour la mise à jour des informations,
3. la distinction entre les modes de raisonnement causal et fondé (basé sur des évidences).

Cette dernière distinction émane directement de l'article de Thomas Bayes en 1763 [Bayes, 1763].

1.2.2 Décomposition d'une distribution de probabilités

Le schéma de base de la décomposition des graphes dirigés acycliques peut être illustré de la façon suivante. Supposons que nous ayons une distribution P définie sur un ensemble de n variables discrètes, ordonnées arbitrairement de cette façon : $X_1, X_2, \ldots, X_n$. La règle de la chaîne permet d'obtenir la décomposition suivante :

$$\begin{aligned} P(X_1, X_2, \ldots, X_n) &= P(X_n|X_{n-1}, \ldots, X_2, X_1) \ldots P(X_2|X_1)P(X_1) && (3.1) \\ &= P(X_1) \prod_{j=n}^{2} P(X_j|X_{j-1}, \ldots, X_1) && (3.2) \end{aligned}$$

Supposons maintenant que les probabilités conditionnelles de certaines variables X_j ne soient pas dépendantes de tous les prédécesseurs de X_j (c'est-à-dire $X_1, X_2, \ldots, X_{j-1}$) mais seulement de certains d'entre eux. En d'autres termes, supposons que X_j soit indépendante de tous ses autres prédécesseurs sauf d'un certain nombre d'entre eux : ceux qui ont une influence directe sur X_j. Nous appellerons cet ensemble restreint $pa(X_j)$. Alors nous pouvons écrire :

$$P(X_j|X_1, \ldots, X_{j-1}) = P(X_j|pa(X_j))$$

et la décomposition 3.2 devient alors :

$$P(X_1, X_2, \ldots, X_n) = \prod_j P(X_j|pa(X_j))$$

Cette formule permet de simplifier énormément les informations nécessaires pour le calcul de la probabilité jointe de l'ensemble $\{X_1, \ldots, X_n\}$. Ainsi, au lieu de spécifier la probabilité de X_j conditionnellement à toutes les réalisations de ses prédécesseurs $X_1, \ldots, X_{j-1}$, seules celles qui sont conditionnées par les éléments de $pa(X_j)$ doivent être précisées. Cet ensemble est appelé *les parents Markoviens de* X_j (ou simplement les *parents*).

Définition 1.1 (Parents Markoviens) *Soit $V = \{X_1, \ldots, X_n\}$ un ensemble ordonné de variables et $P(V)$ la distribution de probabilité jointe sur ces variables. Si $pa(X_j)$ est un ensemble minimal de prédécesseurs de X_j qui rendent X_j indépendant*

de tous ses autres prédécesseurs, alors $pa(X_j)$ sont les parents markoviens *de X_j. En d'autre termes, $pa(X_j)$ est tout sous-ensemble de $\{X_1,\ldots,X_{j-1}\}$ qui satisfait l'équation*

$$P(X_j|pa(X_j)) = P(X_j|X_1,\ldots,X_{j-1}) \tag{3.3}$$

et tel qu'aucun sous-ensemble propre de $pa(X_j)$ ne satisfasse l'équation 3.3.

La définition 1.1 affecte à chaque variable X_j un ensemble $pa(X_j)$ d'autres variables qui sont suffisantes pour déterminer la probabilité de X_j. La connaissance des autres variables est redondante une fois que l'on connaît les valeurs des variables de l'ensemble $pa(X_j)$. Cette affectation de variables peut être représentée par un graphe, dans lequel les noeuds sont les variables et les arcs dirigés dénotent l'influence directe qu'ont les parents sur leurs enfants. Le résultat d'une telle construction est appelé un *réseau bayésien* dans lequel un arc dirigé allant de X_i à X_j définit X_i comme étant un parent markovien de X_j, en accord avec la définition 1.1.

La définition des parents markoviens pose donc la base théorique de la notion de relation modale entre des connaissances dans un réseau bayésien. En effet, il est possible de représenter toute sorte de modalité entre les variables dans un réseau bayésien. Elles peuvent être d'ordre causal, temporel, hiérarchique, etc... En général, une seule modalité est utilisée dans un même réseau et la plupart du temps, il s'agit de la causalité. Ceci permet de représenter l'influence directe d'une variable sur une autre : si il existe un arc dirigé allant d'une variable A à une variable B, alors A est une des causes possibles de B, ou encore A a une influence causale directe sur B.

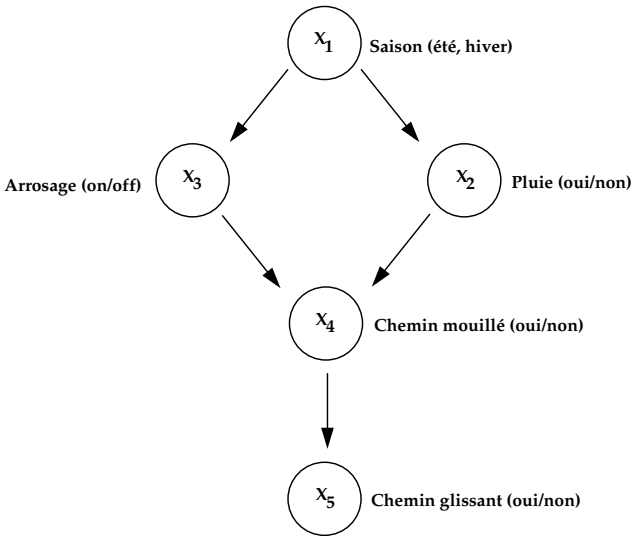

FIG. 3.1 – Un réseau bayésien représentant les dépendances entre cinq variables

La figure 3.1 représente un réseau bayésien simple contenant cinq variables. Il décrit parmi les saisons de l'année (X_1) si la pluie tombe (X_2), si un arrosage est en marche (X_3), si le chemin est mouillé (X_4) et si le chemin est glissant (X_5). Toutes les variables sont binaires. Par exemple, l'absence d'un arc allant de X_1 à X_5 signifie que la saison n'a pas une influence directe sur l'état glissant ou pas du chemin. Autrement dit, le fait que le chemin soit glissant est conditionné par le fait qu'il soit ou non mouillé, rendant inutile la connaissance sur la météo ou l'allumage (ou non) des arrosages, et *a fortiori* sur la saison. Enfin, en reprenant la définition 1.1 (parents markoviens), le graphe de la figure 3.1 se décompose de la façon suivante :

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

Sachant un graphe G et une distribution de probabilités P, alors la décomposition de la définition 1.1 ne nécessite plus d'ordre sur les variables. Nous pouvons conclure qu'une condition nécessaire pour qu'un graphe G soit un réseau bayésien d'une distribution de probabilités P, est que P admette une décomposition sous forme d'un produit dirigé par G tel que [Pearl, 2001] :

$$P(X_1,\ldots,X_n) = \prod_i P(X_i|pa(X_i)) \tag{3.4}$$

Définition 1.2 (Compatibilité de Markov) *Si une fonction de probabilité P admet une factorisation sous la forme de l'équation 3.4 relativement à un graphe acyclique dirigé (GAD) G, on dira que G représente P et que P est compatible ou P est Markov-relatif à G.*

Assurer la compatibilité entre des graphes acycliques dirigés et des probabilités est important en modélisation statistique car la compatibilité est une condition nécessaire et suffisante pour qu'un GAD G puisse expliquer un corpus de données empiriques représentées par P, c'est-à-dire pour décrire un processus stochastique permettant de générer P [Pearl, 1988, Pearl, 2001].

Un moyen facile de caractériser un ensemble de distributions compatible avec un GAD G est de lister l'ensemble des indépendances (conditionnelles) que chacune des distributions devront satisfaire. Ces indépendances peuvent être aisément lues à partir du GAD en utilisant un critère graphique appelé la *d-séparation* (dans [Pearl, 1988], le *d* signifie *directionnel*).

1.2.3 Le critère de *d-séparation*

Considérons trois ensembles disjoints de variables X, Y et Z représentés par trois ensembles de noeuds dans un graphe acyclique dirigé G. Pour savoir si X est indépendant de Y sachant Z dans toute distribution compatible avec G, nous avons besoin de tester si des noeuds correspondants aux variables de Z *bloquent* tous les chemins allant des noeuds de X aux noeuds de Y. Un *chemin* est une séquence consécutive d'arcs (non-dirigés) dans le graphe. Un *blocage* peut être vu comme un arrêt du flux d'informations entre les variables qui sont ainsi connectées. Le flux d'information est dirigé par le sens des arcs et représente le flux des causalités dans le graphe, ou l'ordre dans lequel les influences vont se propager dans le graphe. Cette propagation des influences peut alors être vue comme un envoi d'information d'une variable à ses variables filles.

Définition 1.3 (d-Séparation) *Un chemin p est dit* d-séparé *(ou* bloqué*) par un ensemble Z de noeuds si et seulement si :*

1. *p contient une séquence $i \rightarrow m \rightarrow j$ ou une* divergence *$i \leftarrow m \rightarrow j$ tel que $m \in Z$, ou*
2. *p contient une* convergence *$i \rightarrow m \leftarrow j$ telle que $m \notin Z$ et tel qu'aucun descendant de m n'appartienne à Z.*

Un ensemble Z d-sépare *X de Y si et seulement si Z bloque chaque chemin partant d'un noeud quelconque de X à un noeud quelconque de Y.*

L'idée à la base de la d-séparation est simple quand on attribue une signification aux flèches dans le graphe. Dans la séquence $i \rightarrow m \rightarrow j$ ou dans la divergence $i \leftarrow m \rightarrow j$, si l'on conditionne m (si l'on affecte une valeur à m) alors les variables i et j qui étaient dépendantes conditionnellement à m deviennent indépendantes. Conditionner m *bloque* le flux d'information allant de i à j, c'est-à-dire qu'une nouvelle connaissance sur i ne pourra plus influer m puisque ce dernier est maintenant connu, et donc m ne changeant plus, il n'aura plus d'influence sur j. Donc i, à travers m, n'a plus d'influence sur j non plus. Dans le cas d'une convergence $i \rightarrow m \leftarrow j$, représentant deux causes ayant le même effet, le problème est inverse. Les deux causes sont indépendantes jusqu'à ce qu'on connaisse leur effet commun. Elles deviennent alors dépendantes.

Dans la figure 3.1, si on connaît la saison (X_1) alors X_2 et X_3 deviennent indépendants. Mais si on se rend compte que le chemin est glissant (X_5 est connu) ou qu'il est mouillé (X_4 est connu) alors X_2 et X_3 deviennent dépendants, car réfuter une hypothèse augmentera la probabilité de l'autre (et réciproquement).

Toujours dans la figure 3.1, $X = \{X_2\}$ et $Y = \{X_3\}$ sont d-séparés par $Z = \{X_1\}$ car les deux chemins connectant X_2 à X_3 sont bloqués par Z. Le chemin $X_2 \leftarrow X_1 \rightarrow X_3$ est bloqué car il s'agit d'une convergence dans laquelle le noeud du milieu X_1 appartient à Z. Le chemin $X_2 \rightarrow X_4 \leftarrow X_3$ est bloqué car il s'agit d'une convergence dans laquelle le noeud X_4 et tous ses descendants n'appartiennent pas à Z. Par contre, l'ensemble $Z' = \{X_1, X_5\}$, ne d-sépare pas X et Y : le chemin $X_2 \rightarrow X_4 \leftarrow X_3$ n'est pas bloqué par Z' car X_5 , qui est un descendant du noeud du milieu X_4, appartient à Z'. On pourrait dire que le fait de connaître l'effet X_5 rend ses causes X_2 et X_3 dépendantes. Si l'on observe une conséquence issue de deux causes indépendantes, alors les deux causes deviennent dépendantes l'une de l'autre.

Dans notre exemple, si la pluie est très forte, on pense immédiatement que c'est à cause de la pluie que le chemin est glissant. Donc on en déduit automatiquement que l'arrosage doit être mis hors de cause. De même, si le chemin est vraiment très glissant, on peut en déduire qu'il s'agit là de l'action conjuguée de la pluie et de l'arrosage. Si il est peu glissant, l'arrosage serait plutôt la cause, rendant le fait de pleuvoir quasiment improbable.

Ce schéma de raisonnement est plus connu sous le nom du *paradoxe de Berkson* en statistique [Berkson, 1946].

1.2.4 Quelques propriétés de la *d-séparation*

La connexion entre la *d-séparation* et l'indépendance conditionnelle est établie grâce au théorème suivant que l'on doit à Verma et Pearl dans [Verma and Pearl, 1988] et dans [Geiger et al., 1988] :

Théorème 1.1 (Implications probabilistes de la d-séparation) *Si les ensembles X et Y sont d-séparés par Z dans un GAD G, alors X est indépendant de Y conditionnellement à Z dans chaque distribution compatible (déf. 1.2) avec G. Réciproquement, si X et Y ne sont pas d-séparés par Z dans un GAD G, alors X et Y sont dépendants conditionnellement à Z dans au moins une distribution compatible avec G.*

On notera à présent par $(X \perp\!\!\!\perp Y|Z)_P$ la notion d'indépendance conditionnelle et par $(X \perp\!\!\!\perp Y|Z)_G$ la notion graphique de *d-séparation*. Le théorème peut être réécrit de la façon suivante [Pearl, 2001] :

Théorème 1.2 *Pour tous les ensembles disjoints de noeuds (X,Y,Z) dans un GAD G et pour toute fonction de probabilités P on a :*

1. $(X \perp\!\!\!\perp Y|Z)_G \Longrightarrow (X \perp\!\!\!\perp Y|Z)_P$ *toutes les fois que G et P sont compatibles ; et*
2. *si $(X \perp\!\!\!\perp Y|Z)_P$ est vérifié dans toute distribution compatible avec G, alors $(X \perp\!\!\!\perp Y|Z)_G$.*

Enfin, un autre test de *d-séparation* a été donné dans [Lauritzen et al., 1990]. Il est basé sur la notion de graphes ancestraux. Pour tester si $(X \perp\!\!\!\perp Y|Z)_G$, on efface tous les noeuds de G sauf ceux qui sont dans $\{X,Y,Z\}$ et leurs ancêtres, puis on connecte par un arc chaque paire de noeuds qui a un enfant commun (*mariage* des noeuds) et on transforme le graphe dirigé en graphe non-dirigé. Alors $(X \perp\!\!\!\perp Y|Z)_G$ est vérifié si et seulement si Z intercepte tout chemin allant d'un noeud de X à un noeud de Y dans le graphe non-dirigé résultant. Ici seule la topologie du graphe compte, et non plus l'ordre dans lequel le graphe G avait été construit initialement. Cette approche sera importante lors de la présentation de l'algorithme d'inférence dans les réseaux bayésiens.

2 Modélisation et inférence dans les réseaux bayésiens

2.1 Introduction

Un réseau bayésien permet donc de représenter un ensemble de variables aléatoires pour lesquelles on connaît un certain nombre de relations de dépendances. Appelons U l'ensemble des variables et $P(U)$ la distribution de probabilités sur cet ensemble. Si nous disposons d'une nouvelle information ε sur une ou plusieurs variables, alors on souhaiterait remettre à jour la connaissance que représente le réseau bayésien à travers $P(U)$ à la lumière de cette nouvelle information. Cette remise à jour, qui se fera bien sûr en utilisant la règle de Bayes, est appelée l'inférence. Mathématiquement parlant, l'inférence dans un réseau bayésien est le calcul de $P(U|\varepsilon)$, c'est-à-dire le calcul de la probabilité *a posteriori* du réseau sachant ε.

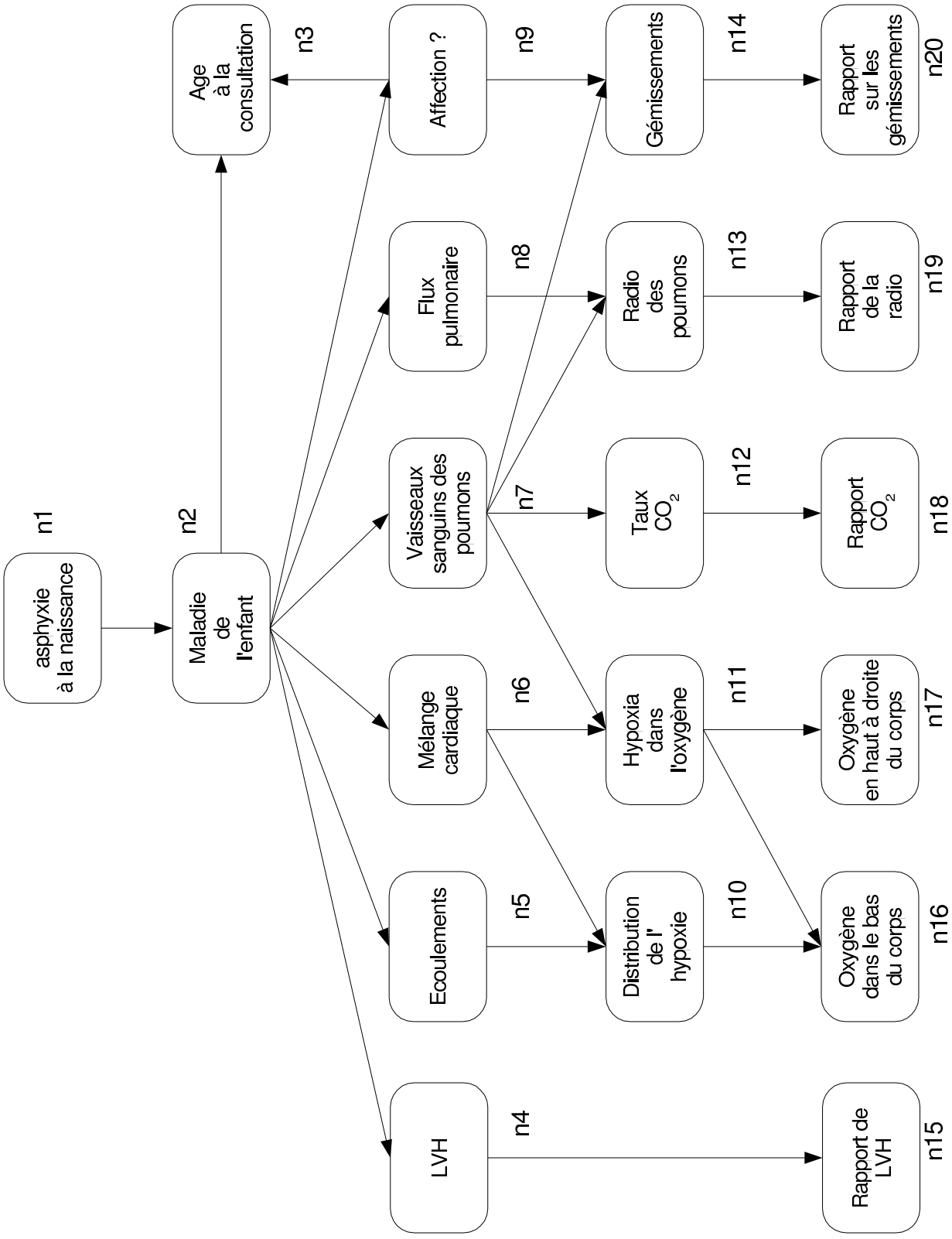

FIG. 3.2 – Représentation du problème d'asphyxie du nouveau-né

2.2 Spécification d'un réseau

2.2.1 Exemple

Pour illustrer notre propos, nous utiliserons un exemple issu de [Cowell et al., 1999]. Il présente un modèle pour le diagnostic de l'asphyxie des nouveaux-nés. Ce domaine médical se prête bien à ce type d'analyse, car sa connaissance clinique est bonne et les données sont disponibles en grande quantité. Nous considérerons que les paramètres cliniques et le diagnostic peuvent être modélisés par des variables aléatoires, et nous aurons donc besoin de spécifier une distribution de probabilités jointe sur ces variables. Ce problème est particulièrement courant dans le domaine des systèmes experts.

La construction d'un tel modèle se décompose en trois étapes distinctes :

1. l'étape *qualitative* : on ne considère ici que les relations d'influence pouvant exister entre les variables prises deux à deux. Ceci emmène naturellement à une représentation graphique des relations entre les variables,
2. l'étape *probabiliste* : elle introduit l'idée d'une distribution jointe définie sur les variables et fait correspondre la forme de cette distribution au graphe créé précédemment.
3. l'étape *quantitative* : elle consiste simplement à spécifier numériquement les distributions de probabilités conditionnelles.

Les maladies cardiaques congénitales peuvent être détectées à la naissance de l'enfant et sont suspectées, en général, par l'apparition de symptômes tels qu'une cyanose (le bébé devient bleu) ou un arrêt ou un dysfonctionnement cardiaque (étouffement de l'enfant). Il est alors vital que l'enfant soit transporté dans un centre spécialisé, et comme l'état de l'enfant peut se détériorer rapidement, un traitement approprié doit être administré avant le transport de l'enfant. Le diagnostic est alors fait en se basant sur les faits cliniques rapportés par le pédiatre, sur une radio, sur un ECG (électro-cardiogramme) et sur une analyse sanguine.

2.2.2 Étape qualitative

Ainsi que J. Pearl le montre dans [Pearl, 1988], l'intérêt des réseaux bayésiens est de permettre aux experts de se concentrer sur la construction d'un modèle qualitatif avant même de penser aux spécifications numériques.

La figure 3.2 [Cowell et al., 1999] représente le réseau bayésien modélisant ce problème de diagnostic médical. Le noeud *Maladie de l'enfant (n_2)* peut prendre six valeurs correspondant aux maladies possibles dans ce cas précis de pathologie. L'arc allant du noeud n_{11} au noeud n_{16} exprime le fait que le taux d'oxygène dans le bas du corps du patient dépend directement de l'oxygène expulsé par le patient (n_{11}) et de la distribution de l'hypoxie dans le corps. De même, l'oxygène expulsé (n_{11}) est directement influencé par l'oxygène qui est dissout dans le corps du patient (n_6) et de l'état des vaisseaux sanguins dans le poumon.

Ce graphe illustre donc la première étape consistant à modéliser qualitativement le problème et à déterminer les influences existant entre les variables.

2.2.3 Étape probabiliste

La spécification probabiliste du modèle passe par une représentation utilisable d'une distribution de probabilités jointe sur l'ensemble des variables. Ainsi qu'il a été montré dans la section 1.2, la décomposition de la distribution de probabilité jointe peut se faire comme dans l'équation 3.4 :

$$P(n_1,\ldots,n_{20}) = \prod_{i=1}^{20} P(n_i|pa(n_i))$$

où les X_i sont les variables représentées par les noeuds du graphe G. La décomposition est toujours la même et permet ainsi de ne spécifier que des probabilités *locales*, c'est-à-dire les probabilités d'une variable sachant uniquement les variables ayant une influence directe sur elle.

2.2.4 Étape quantitative

Cette étape consiste à spécifier les tables de probabilités qui sont, pour tout i, $P(n_i|pa(n_i))$. Une table, c'est la spécification de l'ensemble des probabilités de la variable pour chacune de ses valeurs possibles sachant chacune des valeurs de ses parents. Ces probabilités sont souvent données par un expert du domaine modélisé, ou bien apprise à partir d'un corpus d'exemples. Dans notre exemple, cette étape consiste à spécifier environ 280 valeurs numériques, sachant qu'il y a 20 variables, ayant en moyenne 3 états chacune. Avec des flottants en simple précision, la mémoire nécessaire est d'environ 1,09 Ko. Si nous voulions modéliser ce problème en représentant complètement la distribution de probabilité (donc sans passer par un réseau bayésien), le nombre de valeurs numériques à spécifier serait d'environ 3^{20}, soit 3,5 milliards de valeurs. La mémoire nécessaire serait d'environ 12Go !

2.3 Les principaux algorithmes

2.3.1 Exact et approximatif

Les réseaux bayésiens ont été développés au début des années 1980 pour tenter de résoudre certains problèmes de prédiction et d'abduction, courants en intelligence artificielle (IA). Dans ce type de tâche, il est nécessaire de trouver une interprétation cohérente des observations avec les données connues *a priori*. L'inférence probabiliste signifie donc le calcul de $P(Y|X)$ où X est un ensemble d'observations et Y un ensemble de variables décrivant le problème et qui sont jugées importantes pour la prédiction ou le diagnostic.

Les premiers algorithmes d'inférence pour les réseaux bayésiens sont dans [Pearl, 1982] et [Kim and Pearl, 1983] : il s'agissait d'une architecture à passage de messages et ils étaient limités aux arbres. Dans cette technique, à chaque noeud est associé un *processeur* qui peut envoyer des messages de façon asynchrone à ses voisins jusqu'à ce qu'un équilibre soit atteint, en un nombre fini d'étapes. Cette méthode a été depuis étendue aux réseaux quelconques pour donner l'algorithme JLO qui sera l'objet de notre étude. Cette méthode est aussi appelée *algorithme de l'arbre de jonction* et a été développée dans [Lauritzen, 1988] et [Jensen et al., 1990].

Une autre méthode, développée dans [Pearl, 1988] et dans [Jensen, 1996], s'appelle le *cut-set conditionning* : elle consiste à instancier un certain nombre de variables de manière à ce que le graphe restant forme un arbre. On procède à une propagation par messages sur cet arbre. Puis une nouvelle instanciation est choisie. On réitère ce processus jusqu'à ce que toutes les instanciations possibles aient été utilisées. On fait alors la moyenne des résultats. Dans la figure 3.1, si on instancie X_1 a une valeur spécifique ($X_1 = hiver$, par exemple), alors le chemin entre X_2 et X_3 et passant par X_1 est *bloqué*, et le réseau devient un arbre (cf. figure 3.3) . Le principal avantage de cette méthode est que son besoin en mémoire est

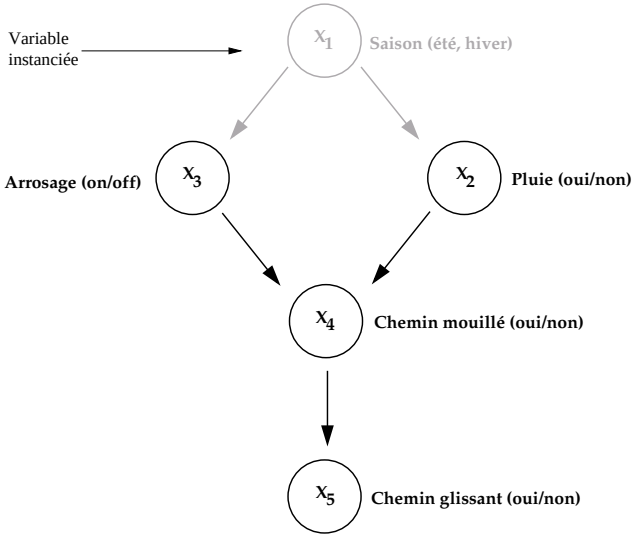

FIG. 3.3 – Graphe de la figure 3.1 transformé en arbre en instanciant X_1.

minimal (linéaire sur la taille du réseau), alors que la méthode de l'arbre de jonction, plus rapide, a une complexité en espace exponentielle. Des méthodes hybrides ont été proposées pour tenter de concilier complexité en temps et en espace, dans [Shachter et al., 1994] et dans [Dechter, 1996b].

Bien que l'inférence dans des réseaux quelconques soit *NP-difficile* [Cooper, 1990], la complexité en temps pour chacune des méthodes citées précédemment est calculable à l'avance. Quand le résultat dépasse une limite raisonnable, on préfère alors utiliser une méthode d'approximation [Pearl, 1988]. Ces méthodes exploitent la topologie du réseau et effectuent un échantillonage de Gibbs sur des sous-ensembles locaux de variables de façon séquentielle et concurrente [Jaakkola and Jordan, 1999], [Jordan et al., 1999].

2.3.2 Approche générale de l'inférence

Soit une distribution P de probabilités, le calcul de $P(Y|X)$ est trivial et nécessite une simple application de la règle de Bayes :

$$P(Y|X) = \frac{P(Y,X)}{P(X)} = \frac{\sum_{h \not< Y \cup X} P(X_H = h, Y, X)}{\sum_{h \not< Y} P(X_H = h, X)}$$

Étant donné que tout réseau bayésien défini aussi une probabilité jointe sur un ensemble de variables aléatoires, il est clair que $P(Y|X)$ peut être calculée à partir d'un GAD G. Le problème de l'inférence se réduit donc à un problème de marginalisation[1] d'une distribution de probabilités jointe. Cependant, le problème ne réside pas tant au niveau du calcul, mais plutôt de son efficacité. En effet, si les variables du GAD G sont binaires, le calcul de $\sum_h P(X_1, \ldots, X_N)$ prendra un temps de $O(2^N)$.

Considérons le réseau bayésien simple de la figure 3.4. Sa probabilité jointe peut être écrite sous la forme :

$$P(A,B,C,D,F,G) = P(A) \cdot P(B|A) \cdot P(C|A) \cdot P(D|B,A) \cdot P(F|B,C) \cdot P(G|F)$$

Supposons que nous voulions marginaliser sur l'ensemble des variables sauf A afin d'obtenir $P(A)$, alors cette marginalisation serait :

$$\sum_{B,C,D,F,G} P(A,B,C,D,F,G) =$$

$$\sum_B P(B|A) \cdot \sum_C P(C|A) \cdot \sum_D P(D|B,A) \cdot \sum_F P(F|B,C) \cdot \sum_G P(G|F)$$

Malgré l'apparente complexité de cette formule, le calcul de la probabilité jointe se résume à des calculs de produits très petits. En prenant la formule de droite à gauche, nous obtenons :

$$\sum_{B,C,D,F,G} P(A,B,C,D,F,G) =$$

$$\sum_B P(B|A) \cdot \sum_C P(C|A) \cdot \sum_D P(D|B,A) \cdot \sum_F P(F|B,C) \cdot \lambda_{G \to F}(F)$$

où $\lambda_{G \to F}(F) = \sum_G P(G|F)$. Dans ce cas précis, $\lambda_{G \to F}(F) = 1$ mais ce n'est pas forcément le cas à chaque fois. L'étape suivante va consister à réduire F de cette façon :

$$\sum_{B,C,D,F,G} P(A,B,C,D,F,G) =$$

$$\sum_B P(B|A) \cdot \sum_C P(C|A) \cdot \lambda_{F \to C}(B,C) \cdot \sum_D P(D|B,A)$$

où $\lambda_{F \to C}(B,C) = \sum_F P(F|B,C) \cdot \lambda_{G \to F}(F)$. En général, on essaie de calculer les termes les plus à gauche possible, de manière à minimiser le nombre de calculs nécessaires. La notation $\lambda_{F \to C}(B,C)$ signifie que l'on fait une sommation sur F et que l'on va ensuite faire une sommation sur C. Ici, C est préféré à B car il est placé plus haut dans l'ordre d'élimination

[1] Si $P(A,B,C,D)$ est une distribution de probabilités sur les variables aléatoires A, B, C et D, alors marginaliser sur D revient, pour chaque valeur de D, à faire la somme des probabilités de cette variable de manière à obtenir $P(A,B,C)$.

des variables. Ainsi, l'ordre dans lequel les variables sont éliminées détermine la quantité de calcul nécessaire pour marginaliser la distribution de probabilités jointe. Cela influe sur la taille nécessaire pour stocker λ. Cet algorithme s'arrête quand on a marginalisé la distribution.

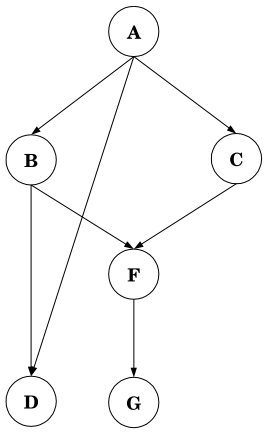

FIG. 3.4 – Un réseau bayésien simple [Kask et al., 2001]

2.4 Algorithme de l'arbre de jonction dit JLO

2.4.1 Introduction

L'algorithme JLO, du nom de ses auteurs : F.V. Jensen, S.L. Lauritzen et K.G. Olesen s'applique à des réseaux ne comprenant que des variables à valeurs discrètes [Lauritzen, 1988],[Jensen et al., 1990]. Des extensions pour des distributions gaussiennes et des mixtures de gaussiennes ont été proposées dans [Lauritzen and Wermuth, 1989] et dans [Cowell et al., 1999]. Un algorithme similaire à été développé par Dawid dans [Dawid, 1992]. Il résout le problème de l'identification du maximum *a posteriori* (MAP) avec une complexité en temps équivalente à celle de l'algorithme JLO. Cet algorithme sera présenté plus tard. De plus, il y a de nombreuses variantes de ces deux algorithmes, mais on peut montrer [Shachter et al., 1994] que tous les algorithmes d'inférence exacte sur les réseaux bayésiens sont équivalents ou peuvent être dérivés de l'algorithme JLO ou de l'algorithme de Dawid. Ainsi, ces algorithmes subsument les autres algorithmes d'inférence exacte dans les modèles graphiques.

L'algorithme se comporte de la façon suivante :

- la phase de *construction* : elle nécessite un ensemble de sous-étapes permettant de transformer le graphe initial en un arbre de jonction, dont les noeuds sont des *clusters* (regroupement) de noeuds du graphe initial. Cette transformation est nécessaire, d'une part pour éliminer les boucles du graphe, et d'autre part, pour obtenir un graphe plus efficace quant au temps de calcul nécessaire à l'inférence, mais qui reste équivalent au niveau de la distribution de probabilité représentée. Cette transformation se fait en trois étapes :
  - la moralisation du graphe,
  - la triangulation du graphe et l'extraction des cliques qui formeront les noeuds du futur arbre,
  - la création d'un arbre couvrant minimal, appelé arbre de jonction ;
- la phase de *propagation* : il s'agit de la phase de calcul probabiliste à proprement parler où les nouvelles informations concernant une ou plusieurs variables sont propagées à l'ensemble du réseau, de manière à *mettre à jour* l'ensemble des distributions de probabilités du réseau. Ceci ce fait en passant des messages contenant une information de mise à jour entre les noeuds de l'arbre de jonction précédemment construit. A la fin de cette phase, l'arbre de jonction contiendra à la distribution de probabilité sachant les nouvelles informations, c'est-à-dire $P(U|\varepsilon)$ où $\underline{U}$ représente l'ensemble des variables du réseau bayésien et ε l'ensemble des nouvelles informations sur lesdites variables. ε peut, par exemple, être vu comme un ensemble d'observations faîtes à partir de capteurs.

Le déroulement de cet algorithme sera illustré sur l'exemple de la figure 3.2.

2.4.2 Moralisation

La première étape de transformation du graphe est la moralisation. Elle consiste à *marier* deux à deux les parents de chaque variable, c'est-à-dire à les relier par un arc non-dirigé. Après avoir moralisé le graphe et introduit des arcs non-dirigés, on finit de transformer complètement le graphe en graphe non-dirigé en enlevant les directions de chaque arc. Si G est le graphe initial, on notera G^m le graphe moralisé. La figure 3.5 montre l'exemple de la section 2.2. Les arcs

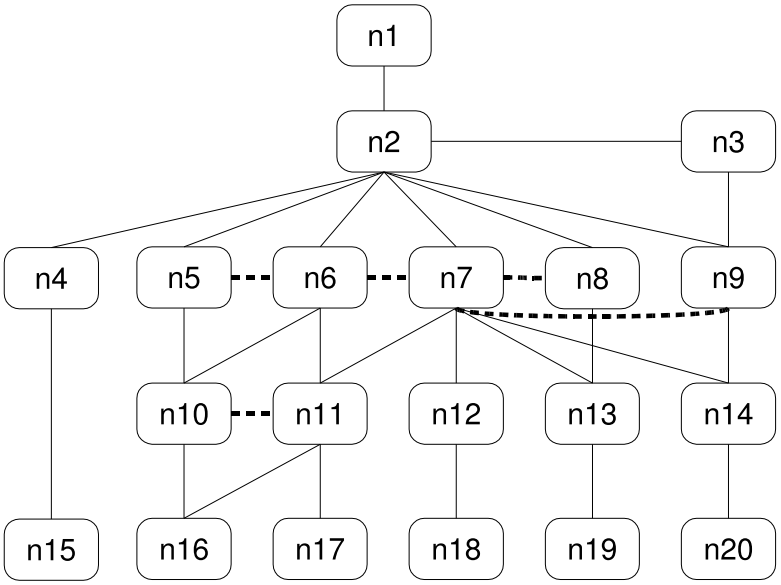

FIG. 3.5 – Graphe moralisé. Les arcs en pointillés ont été rajoutés au cours de la moralisation

en pointillés représentent les arcs qui ont été rajoutés. La moralisation nécessite que tous les noeuds parents d'un même noeud soient reliés deux à deux.

L'idée de base est que la distribution de probabilité satisfasse aux contraintes d'indépendances conditionnelles définies par le graphe G. De plus, [Cowell et al., 1999] montre que le graphe moral G^m satisfait aux mêmes propriétés que G. Cette technique de *moralisation* du graphe permet de révéler toutes les propriétés d'indépendance conditionnelle logiquement impliquées par la factorisation de la distribution jointe. Il s'agit d'une technique équivalente à celle de la *d-séparation* qui aboutit aux mêmes résultats. Cependant, dans le cas de la moralisation, certaines propriétés d'indépendance conditionnelle perdent leur représentation graphique dans le graphe moral G^m. Ces propriétés existent encore mais sont *cachées* dans l'ensemble des distributions de probabilités associées au graphe G^m. [Cowell et al., 1999] présente une justification de l'équivalence de la distribution de probabilités jointe issue du graphe initial G et de la distribution issue du graphe moral G^m.

2.4.3 Triangulation

La deuxième étape consiste à trianguler le graphe moral G^m et à en extraire des cliques de noeuds, qui sont des sous-graphes complets de G. Ces cliques formeront les noeuds de l'arbre de jonction utilisé pour l'inférence. Il faut donc ajouter suffisamment d'arcs au graphe moral G^m afin d'obtenir un graphe triangulé G^T.

L'algorithme de triangulation opère d'une manière très simple. Un graphe est triangulé si est seulement si l'ensemble de ses noeuds peuvent être éliminés. Un noeud peut être éliminé si tous ses voisins sont connectés deux à deux. Donc un noeud peut être éliminé si il appartient à une clique dans le graphe. Une telle clique forme un noeud pour le futur arbre de jonction qui est en train d'être construit. Ainsi, il est possible de trianguler le graphe et de construire les noeuds de l'arbre de jonction en même temps en éliminant les noeuds dans un certain ordre. Si aucun noeud n'est éliminable, il faut en choisir un parmi les noeuds restants et rajouter les arcs nécessaires entre ses voisins pour qu'il devienne éliminable. Le noeud choisi sera celui pour lequel l'espace d'état de la clique formée sera le plus petit possible. En effet, plus les cliques sont petites, plus l'espace de stockage, et *a fortiori* le temps de calcul, sont réduits.

L'efficacité de l'algorithme JLO reste dépendant de la qualité de la triangulation. Mais trouver une bonne triangulation dépend de l'ordre d'élimination des variables. D'une manière générale, trouver une triangulation optimale pour des graphes non-dirigés reste un problème NP-difficile [Yannakakis, 1981]. Dans [Kjaerulff, 1992], Kjaerulff donne un aperçu de plusieurs algorithmes de triangulation pour des graphes acyliques dirigés. Pour des problèmes où les cliques de grande taille sont inévitables, la méthode présentée par Kjaerulff (*simulated annealing*) donne de bons résultats, bien qu'elle nécessite un temps de calcul assez long. Cependant, pour l'algorithme JLO, le calcul de l'arbre de jonction n'est nécessaire qu'une seule fois. Il s'agit là d'un compromis acceptable. Kjaerulff a aussi présenté un algorithme utilisant un *critère d'optimalité* $c(v)$ en fonction d'un noeud v. Par exemple, on peut vouloir maximiser (ou minimiser) une fonction d'utilité ou de coût associée à la sélection d'un noeud du graphe non-dirigé. [Olmsted, 1983] et [Kong, 1986] donnent l'algorithme suivant sur un graphe G ayant k noeuds :

Algorithme 2.1 (Triangulation avec critère d'optimalité) – *Aucun noeud n'est numéroté, $i = k$.*
– *Tant qu'il y a des noeuds non-numérotés faire*
  – *Sélectionner un noeud v non-numéroté optimisant le critère $c(v)$.*
  – *Donner à v le numéro i.*
  – *Former l'ensemble C_i avec le noeud sélectionné et ses voisins non-numérotés.*
  – *Connecter deux à deux tous les noeuds de C_i s'ils ne sont pas encore connectés.*
  – *Éliminer le noeud v sélectionné et décrémenter i de 1.*

Bien sûr, cet algorithme dépend de la qualité du critère d'optimalité $c(v)$ pour sélectionner les noeuds. Ce critère peut tenter de minimiser la taille de l'espace d'états joints de la clique C_i. Ce critère donne en général de bons résultats. On peut aussi tenter de minimiser le nombre d'arcs à ajouter dans une clique C_i si le noeud était sélectionné. L'idée est toujours de privilégier la taille des cliques la plus petite possible afin d'optimiser au mieux le temps de calcul nécessaire aux traitements des tables de probabilités conditionnelles.

La figure 3.6 représente le graphe triangulé de l'exemple de la section 2.2. Dans cet exemple, il a été nécessaire de rajouter deux arcs supplémentaires entre n_5 et n_7 et entre n_5 et n_{11}. Les nombres situés à droite de chaque noeud représentent l'ordre d'élimination des variables au cours de la triangulation. Cet ordre d'élimination va aussi nous permettre d'extraire les cliques. Nous obtenons les cliques suivantes, au cours de la triangulation du graphe G^m :

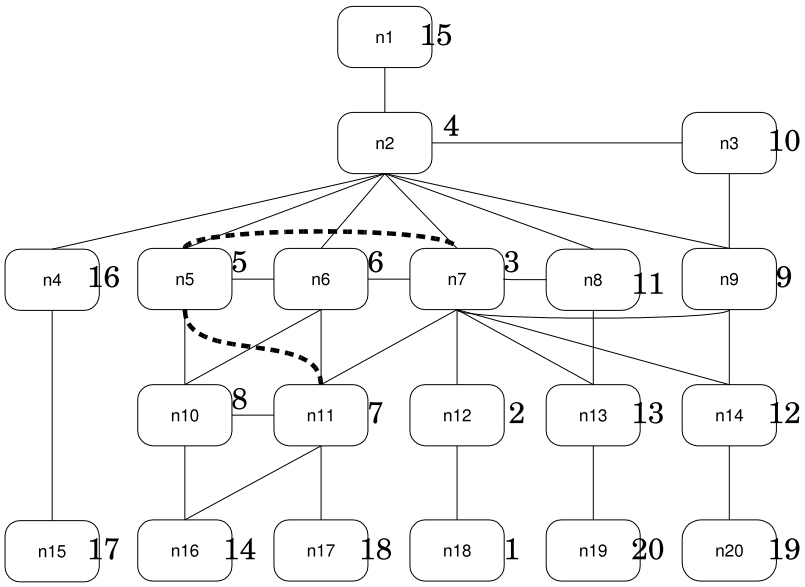

FIG. 3.6 – Graphe triangulé. Les nombres à droite des noeuds représentent l'ordre d'élimination des noeuds. Les lignes en pointillés sont les arcs qu'il a été nécessaire de rajouter

1	n_{12}, n_{18}
2	n_7, n_{12}
3	n_2, n_5, n_6, n_7
4	n_5, n_6, n_7, n_{11}
5	n_5, n_6, n_{10}, n_{11}
6	n_2, n_7, n_9
7	n_2, n_3, n_9
8	n_2, n_7, n_8
9	n_7, n_9, n_{14}
10	n_7, n_8, n_{13}
11	n_{10}, n_{11}, n_{16}
12	n_1, n_2
13	n_2, n_4
14	n_4, n_{15}
15	n_{11}, n_{17}
16	n_{14}, n_{20}
17	n_{13}, n_{19}

Il apparaît clairement que pour toute clique C_i, il existe une clique C_j telle que $C_i \cap C_j \neq \emptyset$. Cette propriété est intéressante et permettra la construction de l'arbre de jonction.

2.4.4 Arbre de jonction

La construction de l'arbre de jonction est la dernière partie avant de procéder à l'inférence proprement dite. Nous rappelons que pour un réseau bayésien donné, l'arbre de jonction est construit une et une seule fois. Les calculs probabilistes auront lieu dans l'arbre de jonction autant de fois que nécessaire. Cependant, pour un réseau bayésien donné, il existe plusieurs arbres de jonction possibles : il sont fonction de l'algorithme de triangulation et de l'algorithme de construction utilisé.

Nous commençons par deux définitions importantes : la décomposition et le graphe décomposable.

Définition 2.1 (Décomposition) *Un triplet (A,B,C) de sous-ensembles disjoints d'un ensemble de noeuds V d'un graphe non-dirigé G forme une* décomposition *de G (ou* décompose *G), si $V = A \cup B \cup C$ et si les conditions suivantes sont satisfaites :*
– *C sépare A de B,*
– *C est un sous-ensemble complet de V.*

A, B ou C peuvent être vides, mais si A et B ne sont pas vides, alors ont dira que l'on a une *décomposition propre* de G.

Définition 2.2 (Graphe décomposable) *Un graphe non-dirigé G est* décomposable *si :*
– *soit il est complet,*
– *soit il possède une décomposition propre (A,B,C) telle que les deux sous-graphes $G_{A\cup C}$ et $G_{B\cup C}$ soit décomposables.*

Ces deux définitions seront utilisées par la suite pour prouver l'existence d'un arbre de jonction. Voici d'abord la définition d'un tel arbre :

Définition 2.3 (Arbre de jonction) *Soit $\mathscr{C}$ une collection de sous-ensembles d'un ensemble fini V de noeuds et soit $\mathscr{T}$ un arbre avec $\mathscr{C}$ comme ensemble de ses noeuds, alors $\mathscr{T}$ est un* arbre de jonction *si toute intersection $C_1 \cap C_2$ d'une paire (C_1,C_2) d'ensembles dans $\mathscr{C}$ est contenue dans chaque noeud sur le chemin unique allant de C_1 à C_2 dans $\mathscr{T}$.*

Si G est un graphe non-dirigé, $\mathscr{C}$ est l'ensemble de ses cliques et $\mathscr{T}$ est un arbre de jonction avec $\mathscr{C}$ son ensemble de noeuds, alors $\mathscr{T}$ est un arbre de jonction (de cliques) pour le graphe G. On a alors le théorème suivant [Cowell et al., 1999] :

Théorème 2.1 *Il existe un arbre de jonction $\mathscr{T}$ de cliques pour le graphe G si et seulement si G est décomposable.*

De plus, l'intersection $S = C_1 \cap C_2$ entre deux voisins dans l'arbre de jonction est aussi un séparateur dans le graphe non-dirigé G des ensembles de noeuds C_1 et C_2 (en fait, il s'agit même d'un séparateur minimal). On appelle S le *séparateur* des noeuds C_1 et C_2 dans l'arbre de jonction. On notera $\mathscr{S}$, l'ensemble des séparateurs. Quand un graphe G admet plusieurs arbres de jonction, on peut alors montrer que $\mathscr{S}$ reste le même, quel que soit l'arbre de jonction. La définition qui suit est utile pour la construction de l'arbre de jonction : il s'agit de la propriété de *running-intersection*.

Définition 2.4 (Propriété de running-intersection) *Une séquence $(C_1, C_2, \ldots, C_k)$ d'ensembles de noeuds a la propriété de* running-intersection *si pour tout $1 < j \leq k$, il existe un $i < j$ tel que $C_j \cap (C_1 \cup \cdots \cup C_{j-1}) \subseteq C_i$.*

Il existe un classement très simple des cliques d'un graphe décomposable : ce classement permet de construire un arbre de jonction possédant la propriété de *running-intersection*. Et inversement, si les cliques ont été ordonnées pour satisfaire cette propriété, alors on peut construire un arbre de jonction avec l'algorithme suivant :

Algorithme 2.2 (Construction de l'arbre de jonction) *Soit un ensemble $(C_1, \ldots, C_p)$ de cliques ordonnées de manière à avoir la propriété de* running-intersection.
- *Associer un noeud de l'arbre de jonction à chaque clique C_i.*
- *Pour $i = 2, \ldots, p$*
  - *ajouter un arc entre C_i et C_j où j est une valeur prise dans l'ensemble $\{1, \ldots, i-1\}$ et tel que :*

$$C_i \cap (C_1 \cup \cdots \cup C_{i-1}) \subseteq C_j$$

L'application de cet algorithme à l'exemple de la section 2.2 nous permet d'obtenir l'arbre de jonction de la figure 3.7.

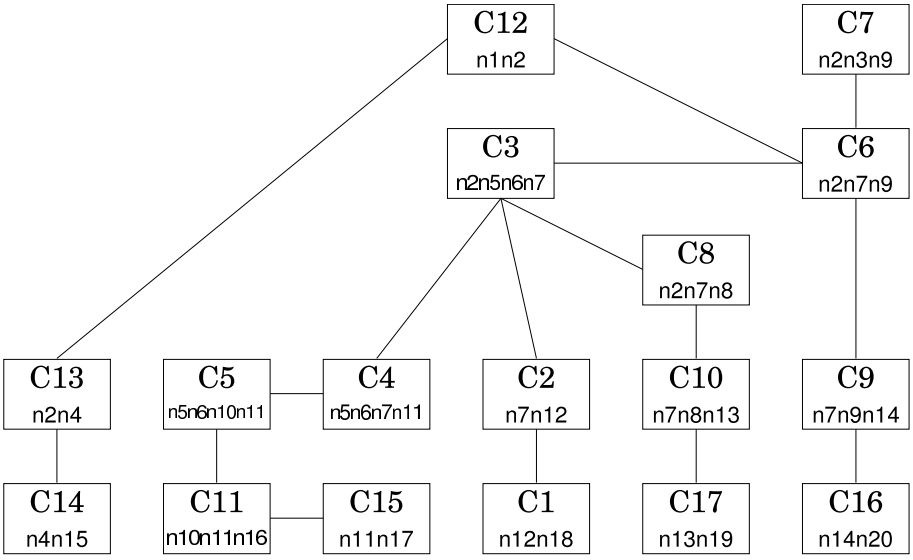

FIG. 3.7 – Arbre de jonction. Les C_i représentent les numéros des cliques, les $n_i \ldots n_j$ sont les noeuds contenus dans chaque clique.

On notera dans cet exemple, que la variable n_2 est contenue dans les cliques C_3, C_6, C_7, C_{12} et C_{13}, avec un particulier $C_3 \cap C_{13} = \{n_2\}$. L'ensemble $\{C_3, C_6, C_7, C_{12}, C_{13}\}$ forme un sous-arbre connecté dans l'arbre de jonction.

Il est aussi possible de construire l'arbre simplement en utilisant l'algorithme de Kruskal [Cormen et al., 1990]. L'algorithme est sensiblement équivalent au précédent. On définit un poids w pour chaque lien entre deux cliques C_i et C_j et tel que

$$w = |C_i \cap C_j|$$

pour tous les couples (i, j). Alors un arbre de cliques satisfera la propriété de *running-intersection* si et seulement si c'est un arbre couvrant de poids maximal [Smyth et al., 1996]. L'algorithme construit un arbre de jonction en choisissant successivement chaque couple de cliques dont le poids est maximal sauf si ce lien crée un cycle.

Ceci termine la construction de l'arbre de jonction. Il est à noter que la complexité dans le pire des cas de l'heuristique de la triangulation est de l'ordre de $O(N^3)$ et que la création de l'arbre (c'est-à-dire le calcul de l'arbre couvrant) est de l'ordre de $O(N^2 \log N)$.

2.4.5 Initialisation de l'arbre de jonction

A partir de cet instant, on considère que l'arbre de jonction est correctement construit. Cette première étape ne doit être faîte qu'une seule fois. Les étapes qui vont maintenant suivre concernent le calcul sur les probabilités dans les réseaux bayésiens. Elles doivent être répétées autant de fois que nécessaire, c'est-à-dire chaque fois que les spécifications numériques changent ou qu'une nouvelle observation est disponible.

Cette étape consiste à utiliser les spécifications numériques du graphe initial G afin de calculer une spécification numérique équivalente pour l'arbre de jonction.

La distribution jointe de probabilité pour un graphe non-dirigé G^u peut être exprimée sous forme d'une simple factorisation :

$$P(X_1, \ldots, X_n) = \prod_{C \in \mathscr{C}} a_C(x_C) \tag{3.5}$$

où $\mathscr{C}$ est l'ensemble des cliques de G^u, x_C est une affectation de valeurs aux variables de la clique C et les fonctions a_C sont des fonctions non-négatives, prenant leurs valeurs dans l'ensemble des affectations possibles des valeurs des variables de la clique C et rendent une valeur dans l'intervalle $[0, \infty[$. L'ensemble des fonctions de cliques associées à un graphe non-dirigé G^u représentent la spécification numérique de ce graphe [Smyth et al., 1996]. Dans la littérature sur les champs de Markov, une telle fonction $a_c(x_C)$ est souvent appelée une fonction de potentiel.

Nous savons d'ores et déjà que les cliques du graphe triangulé ont été arrangées sous forme d'un arbre de jonction. En considérant à présent l'ensemble des séparateurs associés à chaque couple de cliques adjacentes dans l'arbre de jonction, on donne à chaque séparateur une fonction de potentiel b_S définie de façon équivalente aux fonctions de potentiel a_C. Comme un séparateur est égal à l'intersection de deux cliques adjacentes, la distribution de probabilités jointe du réseau bayésien initial peut se factoriser de la façon suivante :

$$P(X_1 \ldots X_n) = \frac{\prod_{C \in \mathscr{C}} P(x_C)}{\prod_{S \in \mathscr{S}} P(x_S)} \tag{3.6}$$

où $P(x_C)$ et $P(x_S)$ sont les distributions de probabilités marginales jointes des variables de la clique C (respectivement du séparateur S). Ce résultat est très important et permet de justifier le calcul d'inférence dans les réseaux bayésiens avec l'algorithme de l'arbre de jonction.

A partir de cette nouvelle formulation de la distribution de probabilité d'un réseau bayésien, l'initialisation se fait très simplement. Comme l'arbre de jonction vérifie la propriété de *running-intersection,* on sait alors que chaque variable X_i se trouve dans au moins une clique. On affecte alors, de façon unique, chaque X_i à une et une seule clique de l'arbre. Certaines cliques risquent de n'avoir aucune variable affectée. Après avoir affecté l'ensemble des variables, chacune à une clique particulière, on définit les fonctions de potentiel de la façon suivante :

$$a_C(x_C) = \begin{cases} \prod_i P(X_i | pa(X_i)) \, si \, X_i \, est \, mis \, dans \, C \\ 1 \, si \, aucune \, variable \, n'est \, dans \, C \end{cases}$$

Les fonctions de potentiel b_S des séparateurs sont initialisées à 1.

A cet instant, l'arbre de jonction est initialisé et consistant avec le réseau bayésien initial. On peut donc propager une observation et calculer la probabilité *a posteriori* de chaque variable du réseau sachant des observations.

2.4.6 Propagation par passages de messages locaux

Le principe de l'algorithme est de passer l'information nouvelle d'une clique à ses voisines dans l'arbre de jonction, et de mettre à jour les voisines et les séparateurs avec cette information locale. Le point important dans l'algorithme JLO est que la représentation $P(X_1 \ldots X_n)$ de l'équation 3.6 reste vraie après chaque passage de messages d'une clique à une autre. Une fois que tous les messages locaux ont été transmis, la propagation convergera vers une représentation marginale de la distribution de probabilités sachant le modèle initial (c'est-à-dire les paramètres du réseau bayésien qui nous ont permis d'initialiser l'arbre de jonction) et sachant les évidences observées.

Quelques définitions

On appelle table de probabilités conditionnelles (CPT : conditional probability table), le tableau contenant l'ensemble des probabilités d'un ensemble de variables aléatoires discrètes pour chacune des valeurs de ces variables. Un tel tableau forme un *potentiel discret* ou simplement un *potentiel*. De plus, la table de probabilités conditionnelles représentant $P(x|pa(x))$ forme aussi un *potentiel* avec la propriété additionnelle de sommer à 1 pour chaque configuration des parents.

Une *évidence* correspond à une information nous donnant avec une certitude absolue la valeur d'une variable. Dans le cas normal, un noeud X binaire d'un réseau bayésien contiendra l'information *« je pense que $X = x_1$ avec une certitude de $P(X = x_1)$ et je pense que $X = x_2$ avec une certitude de $P(X = x_2)$. »* Dans le cas d'une évidence, le noeud contiendra *« je sais que $X \neq x_2$. »* Une deuxième forme d'évidence est appelée évidence vraisemblable ou simplement vraisemblance (*likelihood findings*), et permet d'apporter une observation avec simplement une distribution de vraisemblance sur l'ensemble des états possibles de l'observation, relatant l'incertitude que l'on a sur l'observation. La somme des valeurs doit être égale à 1.

Flux d'information entre les cliques

L'équation 3.6 nous permet de spécifier numériquement les paramètres de l'arbre de jonction :

$$P(U) = \frac{\prod_{C \in \mathscr{C}} a_C(x_C)}{\prod_{S \in \mathscr{S}} b_S(x_S)}$$

où a_C et b_S sont des fonctions de potentiel non-négatives.

Le passage de messages procède de la façon suivante. On définit le *flux* d'une clique C_i à une clique adjacente C_j de la manière suivante. Soit S_k le séparateur de ces deux cliques, alors

$$b^*_{S_k}(x_{S_k}) = \sum_{C_i \setminus S_k} a_{C_i}(x_{C_i}) \tag{3.7}$$

est la marginalisation sur l'ensemble des états des variables qui sont dans la clique C_i mais pas dans le séparateur S_k. La clique C_j est mise à jour avec le potentiel suivant :

$$a^*_{C_j}(x_{C_j}) = a_{C_j}(x_{C_j})\lambda_{S_k}(x_{S_k})$$

où

$$\lambda_{S_k}(x_{S_k}) = \frac{b^*_{S_k}(x_{S_k})}{b_{S_k}(x_{S_k})}$$

Le terme $\lambda_{S_k}(x_{S_k})$ est appelé le *facteur de mise à jour*. L'idée du message est de transférer la nouvelle information que C_i a reçu (l'évidence) et que C_j ne connaissait pas encore. Cette nouvelle information est alors *résumée* dans le séparateur S_k qui est le seul point commun entre C_i et C_j. Un flux correspond à l'envoi de messages depuis C_i vers tous ses voisins dans l'arbre de jonction. Ce flux introduit alors une nouvelle représentation de $P(U)$ probabiliste de l'arbre de jonction telle que :

$$K^* = (\{a^*_C : C \in \mathscr{C}\}, \{b^*_S : s \in \mathscr{S}\})$$

Enfin, pour compléter l'algorithme, il faut un ordonnancement du passage des messages. Cet ordonnancement est dicté par l'arbre de jonction. Un ordonnancement est l'ordre dans lequel les messages sont passés d'une clique à une autre de telle manière que toutes les cliques reçoivent une information de chacune de leur voisine.

L'ordonnancement le plus direct opère en deux temps. On choisit une clique comme racine de l'arbre de jonction. Tout noeud de l'arbre peut être choisi comme racine. Ensuite la première phase dite de *collection* consiste à passer les messages depuis les feuilles de l'arbre jusqu'à la racine. Si un noeud doit recevoir plusieurs messages, alors les messages sont envoyés séquentiellement. L'ordre n'est pas important. Une fois la phase de collection complétée, commence la phase de *distribution* qui consiste à opérer de manière inverse : les messages sont transmis depuis la racine jusqu'aux feuilles. Il y a au plus deux messages parcourant chaque arc de l'arbre : celui du fils et celui du père. On notera en plus que le flux des messages dans l'arbre de jonction n'a aucun lien avec les arcs du réseau bayésien initial et ne reflètent pas la structure du réseau.

A la fin, le réseau a atteint un *état d'équilibre*, ce qui signifie que si aucune information nouvelle n'est introduite dans le réseau, alors un nouveau passage de messages dans l'arbre de jonction ne modifiera pas les fonctions de potentiel. Ceci est cohérent avec le fait que le passage d'un message correspond à la transmission d'une nouveauté (en terme d'information probabiliste) d'une clique à une autre. Une fois que toutes les nouvelles informations ont été propagées à l'ensemble du réseau, l'ensemble des noeuds a eu connaissance de l'ensemble des nouveautés. L'état d'équilibre est atteint.

Si ε est l'ensemble des évidences introduites dans le réseau, et si $P(U)$ est la distribution de probabilité *a priori* du réseau, alors l'algorithme JLO, tel qu'il a été présenté, donne après propagation, $P(U|\varepsilon)$.

Entrer une évidence dans le réseau

Classiquement, un réseau bayésien est utilisé de façon dynamique. A chaque fois qu'une nouvelle information est obtenue, elle est insérée dans le réseau et on la propage à l'ensemble du réseau. Avant une phase de propagation, JLO permet d'insérer autant d'évidences qu'il y a de variables.

D'un point de vue plus formel, une *évidence* est une fonction $\varepsilon : \chi \rightarrow \{0,1\}$. Cette évidence représente le fait que certains états χ de la variable aléatoire sont *impossibles*. Si tous les états sauf un sont déclarés impossibles, alors après propagation de l'évidence, la variable ayant reçu l'évidence aura une probabilité $P(X_\varepsilon = x_i) = 1$ où x_i représente le seul état possible. Une évidence vraisemblable est une fonction $\varepsilon : \chi \rightarrow [0,1]$ et telle que la somme des affectations à chaque état de la variable soit égale à 1.

Pour insérer une évidence dans un réseau bayésien, on procède de la façon suivante : pour chaque clique contenant la variable v recevant l'évidence, la fonction de potentiel ϕ_C de la clique (autrement dit sa table de probabilités) est multipliée par l'évidence. En pratique, s'il s'agit d'une évidence simple (au contraire d'une évidence semblable), alors on met à 0 les entrées de la table correspondant aux valeurs impossibles décrétées par l'évidence. Le fonction de potentiel modifiée correspond alors à la représentation de $P^\varepsilon(x) \equiv P(x\&\varepsilon) \equiv P(x)\varepsilon(x) \propto P(x|\varepsilon)$.

Après avoir entré une (ou plusieurs) évidence(s) dans le réseau, on procède à une propagation complète (collection et distribution) de manière à ce que le réseau atteigne un état d'équilibre. Ensuite l'ensemble des tables de probabilités des cliques de l'arbre de jonction sont normalisées. On obtient la formule suivant pour la probabilité jointe *a posteriori* :

$$P(x\&\varepsilon) = \frac{\prod_{C\in\mathscr{C}} P(x_C\&\varepsilon)}{\prod_{S\in\mathscr{S}} P(x_S\&\varepsilon)}$$

et en normalisant on obtient finalement :

$$P(x|\varepsilon) = \frac{\prod_{C\in\mathscr{C}} P(x_C|\varepsilon)}{\prod_{S\in\mathscr{S}} P(x_S|\varepsilon)}$$

A ce moment, obtenir la probabilité *a posteriori* sachant l'évidence d'une variable quelconque, revient à prendre une clique contenant ladite variable et à marginaliser la table de probabilités afin d'obtenir la table de probabilité de la variable seule. Toute clique contenant la variable d'intérêt est un candidat adéquat.

2.4.7 Un exemple de propagation

Pour illustrer l'algorithme de propagation, nous allons réutiliser l'exemple de la figure 3.2. Considérons deux cliques adjacentes $C_{13} = \{n_2, n_4\}$ et $C_{14} = \{n_4, n_{15}\}$ et leur séparateur $S = \{n_4\}$.

L'arbre de jonction (dont on ne représente qu'une petite partie ici) est initialisé avec $P(n_{15}|n_4)$ pour C_{14} et $P(n_4|n_2)$ pour C_{13} (voir figure 3.8(a)). La variable n_2 servira, quant à elle, à initialiser la clique C_{12} avec $P(n_2|n_1)$. Les séparateurs sont initialisés à 1.

Ensuite on incorpore l'évidence $rapport\,LVH = oui$, et la clique C_{14} contient alors des 0 correspondant aux états impossibles. Après passage du message, le séparateur et la clique C_{13} sont mis à jour, ce qui donne les potentiels de la figure 3.8(b). Deux étapes ont été nécessaires :

– on marginalise C_{14} sur toutes les variables non contenues dans S (en fait toutes sauf n_{15}) pour obtenir b_S^*, le nouveau potentiel du séparateur (3.8(b)) ;

– on calcule le ratio λ qui sert à modifier le potentiel de la clique C_{13}, en multipliant chaque terme de ce potentiel avec λ.

Dans la figure 3.8(c), on retrouve C_{13} modifié par les messages en provenance du reste de l'arbre de jonction (phase de distribution). Dans cette figure, le séparateur et C_{14} n'ont pas encore reçu le message de C_{13}. On calcule donc, de la même manière, le message de C_{13} à C_{14} et on modifie le séparateur, puis on remet à jour C_{14}. La figure 3.8(d) nous donne les nouveaux potentiels pour C_{14} et le séparateur. A ce moment là, la propagation est complètement finie en terme d'envoi de messages entre les cliques. Le potentiel de chaque clique est maintenant égal à $P(C_i, \varepsilon)$, c'est-à-dire, la probabilité de la clique *et* de l'évidence. En normalisant la clique sur ε, on obtient $P(\varepsilon) = 0.284$. Ceci est vrai pour toute clique. On peut alors normaliser sur l'ensemble des cliques afin d'obtenir la probabilité du réseau sachant l'évidence (et non plus en conjonction avec l'évidence). Ceci nous donne finalement la figure 3.8(e), et termine complètement l'algorithme JLO.

On peut bien sûr insérer plus d'une évidence à la fois, et propager après pour calculer $P(U|\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)$. On peut aussi incorporer les évidences une à la fois et propager à chaque fois pour voir l'évolution de la probabilité du réseau. Dans les deux cas, on aura le même résultat à la fin. De plus, si une variable contient n états discrets, il est possible de donner au plus $n-2$ états impossibles, laissant ainsi un certain *degré de liberté* sur les états probables de la variable.

2.4.8 Complexité de l'étape de propagation

La complexité de l'algorithme JLO au moment de la propagation des messages est de $O(\sum_{i=1}^{N_C} n_e(C_i))$ où N_C est le nombre de cliques de l'arbre de jonction et $n_e(C_i)$ est le nombre d'états de la clique C_i. Ainsi, pour réduire cette complexité, il est nécessaire de construire des cliques ayant un petit nombre de variables (et si possible, avec des variables ayant un petit nombre d'états). Cependant, le problème de trouver un arbre de jonction optimal, avec des cliques les plus petites possible, reste un problème NP-difficile. En pratique, l'heuristique proposée dans [Jensen et al., 1990] donne de bons résultats.

3 Conclusion

3.1 Max-propagation

L'algorithme JLO est à la base d'une famille plus complète d'algorithmes permettant de faire de l'inférence sachant un ensemble d'évidences sur certaines variables du réseau. En particulier, l'algorithme de Dawid sert à trouver la configuration la plus probable de toutes les variables sachant une représentation sous forme de réseau bayésien d'une fonction p de probabilité [Dawid, 1992]. L'algorithme JLO est utilisé en modifiant simplement la routine principale de propagation. Pour créer les messages inter-cliques, au lieu de faire une marginalisation avec une somme , on peut utiliser une fonction de maximisation telle que, si $W \subseteq U \subseteq V$ sont des sous-ensembles des noeuds du réseau et ϕ est un potentiel sur U, alors l'expression $M_{U\setminus W}\phi$ dénote la *max-marginalisation* de ϕ sur W, définie par

$$\left(M_{U\setminus W}\phi\right)(x) = \max_{z \in \mathscr{U}\setminus\mathscr{W}} \phi(z.x)$$

FIG. 3.8 – Exemple d'une propagation entre les cliques C_{14} et C_{13} (et le reste du réseau)[Cowell et al., 1999]

Les flux de messages sont alors calculés de la même façon qu'avec l'algorithme JLO, en remplaçant simplement la marginalisation de la formule 3.7 par celle donnée précédemment. Le schéma de propagation est ensuite exactement le même. Les mêmes résultats et les mêmes propriétés que ceux de l'algorithme JLO s'appliquent à l'algorithme de Dawid. Après une propagation complète, le réseau contient la configuration jointe la plus probable de l'ensemble des variables du réseau. Bien sûr, si une variable a été *instanciée* par une évidence, elle gardera sa valeur, après propagation. Cet algorithme peut être vue comme donnant *la meilleure explication* de l'évidence.

3.2 Perspectives

Même si le problème de l'inférence est résolu, un algorithme tel que JLO n'est applicable que sur des applications raisonnables. Il est possible de traiter de très grands réseaux, à condition que les cliques gardent une taille acceptable[2]. Les recherches actuelles s'orientent vers l'amélioration des divers algorithmes de propagation. On trouvera des références dans [Kjaerulff, 1998], [Shenoy, 1997] ou encore [Madsen and Jensen, 1998].

L'algorithme de Viterbi pour les modèles de Markov cachés [Viterbi, 1967] est un cas particulier de l'algorithme de max-propagation [Smyth et al., 1996], comme d'autres algorithmes de décodage tels que BCJR [Bahl et al., 1974] (dans [Frey, 1998] on trouvera plus de détails sur cette méthode). On peut aussi transformer d'autres algorithmes en instance des algorithmes de propagation [Murphy, 2002] tels que les transformations de Hadamard, les FFT [Kschischang et al., 2001], les algorithmes de satisfaction en logique propositionnelle (Davis et Putnam) [Dechter, 1998] ou encore certains algorithmes de parsing de grammaires [Parsing, 1999].

L'algorithme de propagation (JLO, Dawid, etc...) a été généralisé par R. Dechter en 1996. Il porte le nom d'algorithme de *bucket élimination* [Dechter, 1996a].

Cependant, les recherches ne se limitent pas à la découverte d'algorithmes de propagations plus performants, mais s'orientent aujourd'hui vers la modélisation efficace et intuitive de réseaux bayésiens toujours plus grands :
- apprentissage de la structure du réseau [Friedman, 1998],
- adaptation en ligne de la structure d'un réseau [Chaodhury et al., 2002],
- évaluation de la pertinence d'un réseau par rapport à un problème donné,
- modélisation intuitive avec de nouveaux modèles comme les réseaux bayésiens orientés objet [Koller and Pfeffer, 1997], les réseaux bayésiens hiérarchiques [Murphy and Paskin, 2001], etc...
- incorporation de la notion de temps dans un réseau bayésien [Aliferis and Cooper, 1995], [Jr. and Young, 1999].

Bien que des solutions partielles existent déjà, ces problèmes, encore aujourd'hui, restent largement ouverts et dignes d'intérêt.

[2] La *taille acceptable* dépend du nombre de variables et du nombre de valeurs discrètes que peut prendre chaque variable. Par exemple, si toutes les variables sont binaires, alors une taille acceptable ne dépassera pas 20 à 25 variables (entre 4Mo et 128Mo par table de probabilités).

Bibliographie

[Abidi et al., 1992] Abidi, Mongi, A., Gonzalez, Rafael, C., and Elfes, A. (1992). *Data fusion in robotics and machine intelligence*, chapter 3, pages 137–163. Academic Press.

[Abidi and Gonzalez, 1992] Abidi, M. and Gonzalez, R. (1992). *Data Fusion in Robotics and Machine Intelligence*. Academic Press.

[Aliferis and Cooper, 1995] Aliferis, C. and Cooper, G. (1995). A structurally and temporally extended bayesian belief network model : Definitions, properties and modeling techniques. In *UAI*.

[Arnborg et al., 2000] Arnborg, S., Brynielson, J., Artman, H., and Wallenius, K. (2000). Information awareness in command and control : Precision, quality, utility. In *Fusion 2000*.

[Ayari, 1996] Ayari, I. (1996). *Fusion multi-capteurs dans un cadre multi-agents : application à un robot mobile*. PhD thesis, Université Henri Poincaré - Nancy I.

[Bahl et al., 1974] Bahl, L., Cocke, J., Jelinek, F., and Raviv, J. (1974). Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Transactions on Information Theory*, 20 :284–287.

[Baldi and Brunak, 1998] Baldi, P. and Brunak, S. (1998). *Bioinformatics, the Machine Learning Approach*. MIT Press.

[Bar-Shalom and Li, 1995] Bar-Shalom, Y. and Li, X.-R. (1995). *Multitarget-Multisensor Tracking : Principles and Techniques*. Number 3rd printing. YBS.

[Barret, 1990] Barret, I. (1990). *Synthèse d'algorithmes de poursuite multi-radars d'avions civils manouevrants*. PhD thesis, Ecole Nationale supérieure de l'aeronautique et de l'espace.

[Bayes, 1763] Bayes, T. (1763). A essay toward solving a problem in the doctrine of chance. *Philosophical Transactions of the Royal Society*, 53 :370,418.

[Becker and Naïm, 1999] Becker, A. and Naïm, P. (1999). *Les réseaux bayésiens*. Eyrolles, eyrolles edition.

[Bellot et al., 2002a] Bellot, D., Boyer, A., and Charpillet, F. (2002a). Designing smart agent based telemedicine systems using dynamic bayesian networks : an application to kidney disease people. In *Proc. HealtCom 2002*, pages 90–97, Nancy, France.

[Bellot et al., 2002b] Bellot, D., Boyer, A., and Charpillet, F. (2002b). A new definition of qualified gain in a data fusion process : application to telemedicine. In *Fusion 2002*, Annapolis, Maryland, USA.

[Berkson, 1946] Berkson, J. (1946). Limitations of the application of fourfold table analysis to hospital data. *Biometrics Bulletin*, 2 :47–53.

[Beschta et al., 1993] Beschta, A., Dressler, O., Freitag, H., Montag, M., and Struss, P. (1993). Dpnet-a second generation expert system for localizing faults in power transmission networks. In *Proceedings International Conference on Fault Diagnosis (Tooldiag-93)*, pages 1019–1027, Toulouse, France.

[Bigi, 2000] Bigi, B. (2000). *Contribution à la modélisation du langage pour des applications de recherche documentaire et de traitement de la parole*. PhD thesis, Université d'Avignon et des Pays du Vaucluse, Avignon, France.

[Bloch and Maître, 1994] Bloch, I. and Maître, H. (1994). Fusion de données en traitement d'images : modèles d'information et décisions. *Traitement du Signal*, 11(6) :435–446.

[Brogi et al., 1988] Brogi, A., Filipi, R., Gaspari, M., and Turini, F. (1988). An expert system for data fusion based on blackboard architecture. In *Proc. 8th Int. Workshop Expert Systems and their Applications*, pages 147–165, Avignon, France.

[Buchanan and Shortliffe, 1984] Buchanan, B. and Shortliffe, E. (1984). *Rule-based Expert Systems : the MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, MA.

[Chandrasekaran, 1988] Chandrasekaran, B. (1988). Generic tasks as building blocks for knowledge-based systems : the diagnosis and routine examples. *The Knowledge Engineering Review*, 3 :183–210.

[Chanliau et al., 2001] Chanliau, J., Charpillet, F., Durand, P., Hervy, R., Pierrel, J., Romary, L., and Thomesse, J. (2001). Un systÃšme de tlÃ©surveillance de maladea domicile. Brevet français n 00 00903.

[Chaodhury et al., 2002] Chaodhury, T., Pentland, A., Regh, J., and Pavlovic, V. (2002). Boosted learning in dynamic bayesian networks for multimodal detection.

[Cheng and Bell, 1997] Cheng, J. and Bell, D. (1997). Learning bayesian networks from data : an efficient approach based on information theory. In *Proceeding of the sixth ACM International Conference on Information and Knowledge Management*.

[Cooper, 1990] Cooper, G. (1990). Computational complexity of probabilistic inference using bayesian belief networks. *Artificial Intelligence*, 42 :393–405.

[Cooper and Herskovitz, 1992] Cooper, G. and Herskovitz, E. (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9 :309–347.

[Cormen et al., 1990] Cormen, T., Leiserson, C., and Rivest, R. (1990). *Introduction à l'algorithmique*.

[Cowell et al., 1999] Cowell, R., Dawid, A., Lauritzen, S., and Spiegelhalter, D. (1999). *Probabilistic Networks and Expert Systems*. ISBN : 0-387-98767-3.

[Cox and Wermuth, 1996] Cox, D. and Wermuth, N. (1996). *Multivariate Dependancies - Models, Analysis and Interpretation*. Chapman Hall, London.

[Crowley and Demazeau, 1993] Crowley, J. and Demazeau, Y. (1993). Principles and techniques pour sensor data fusion. *Signal processing*, 32 :5–27.

[Dague, 1994] Dague, P. (1994). Model-based diagnosis of analog electronic circuits. 11 :439–492.

[Dagum and Galper, 1995] Dagum, P. and Galper, A. (1995). Time-seris prediction using belief network models. *Intl. Journal of Human-Computer Studies*, 42 :617–632.

[Daoudi et al., 2000] Daoudi, K., Fohr, D., and Antoine, C. (2000). A new approach for multi-band speech recognition based on probabilistic graphical models. In *ICSLP'2000*, Beijing, China.

[Dawid, 1992] Dawid, A. (1992). Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, 2 :25–36.

[Dechter, 1996a] Dechter, R. (1996a). Bucket elimination : a unifying framework for probabilistic inference. In Horvitz, E. and Jensen, F., editors, *Proceedings of the 12th Annual Conference on Uncertainty in Artificial Intelligence*, pages 211–219, San Francisco, California. Morgan Kaufman.

[Dechter, 1996b] Dechter, R. (1996b). Topological parameters for time-space tradeoff. In Horvitz, E. and Jensen, F., editors, *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, pages 220–227, San Francisco. Morgan Kaufman.

[Dechter, 1998] Dechter, R. (1998). *Bucket elimination : a unifying framework for probabilistic inference*. MIT Press.

[Delone and McLean, 1992] Delone, W. and McLean, E. (1992). Information systems success : the quest for the dependant variable. *Information Systems Research*, 3 :60–95.

[Dempster, 1967] Dempster, A. (1967). Upper and lower probabilities induced by a multivalued mapping. *Ann. Math. Stat.*, pages 325–339.

[Downing, 1993] Downing, K. (1993). Physiological applications of consistency-based diagnosis. *Artificial Intelligence in Medicine*, 5 :9–30.

[Durand and Kessler, 1998] Durand, P.-Y. and Kessler, M. (1998). *La dialyse péritonéanle automatisée*.

[Elfes, 1989] Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *IEEE Computer*, 6 :46–57.

[Fine et al., 1998] Fine, S., Singer, Y., and Tishby, N. (1998). The hierarchical hidden markov model : Analysis and applications. *Machine Learning*, pages 32–41.

[Frey, 1998] Frey, B. (1998). *Graphical Models for Machine Learning and Digital Communications*. Cambridge, Massachusetts.

[Friedman, 1998] Friedman, N. (1998). The Bayesian structural EM algorithm. In Kaufmann, M., editor, *Proc. Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI '98)*, pages 129–138, San Francisco, CA.

[Gales, 1999] Gales, M. (1999). Semi-tied covariance matrices for hidden markov models. *IEEE Transaction on Speech and Audio Processing*, 7 :272–281.

[Gebhardt and Kruse, 1998] Gebhardt, J. and Kruse, R. (1998). Information Source Modelling for Consistent Data Fusion. In Hamid R. Arabnia and Dongping (Daniel) Zhu, editors, *Proceedings of the International Conference on Multisource-Multisensor Information Fusion - Fusion'98*, volume I, pages 27–34, Las Vegas, Nevada, USA. CSREA Press.

[Geiger et al., 1988] Geiger, D., Verma, T., and Pearl, J. (1988). Identifying independance in bayesian networks. *Networks*, 20 :507–534.

[Hall and Llinas, 1997] Hall, D. and Llinas, J. (1997). An introduction to multisensor data fusion. In IEEE, editor, *Proceedings of the IEEE*, volume 85, pages 6–23.

[Hamilton, 1994] Hamilton, J. (1994). *Time Series Analysis*.

[Haton et al., 1998] Haton, J., Charpillet, F., and Haton, M. (1998). Numeric/symbolic approaches to data and information fusion. In *Proceedings of the International Conference on Multisource-Multisensor Information Fusion - Fusion'98*, volume II, pages 888–895, Las Vegas, Nevada, USA. CSREA Press.

[Heckerman and et al., 1995] Heckerman, D. and et al., D. G. (1995). Learning bayesian networks : the combination of knowledge and statistical data. *Machine Learning*, 20 :197–243.

[Hertz and Krogh, 1991] Hertz, J. and Krogh, A. (1991). Palmer : Introduction to the theory of neural computation.

[Isham, 1981] Isham, V. (1981). An introduction to spatial point processes and markov random fields. *International Statistical Review*, 49 :21–43.

[Jaakkola and Jordan, 1999] Jaakkola, T. and Jordan, M. I. (1999). Variational probabilistic inference and the QMR-DT network. *Journal of Artificial Intelligence Research*, 10 :291–322.

[Jeanpierre, 2002] Jeanpierre, L. (2002). *Apprentissage et adaptation pour la modélisation stochastique de systèmes dynamiques réels*. PhD thesis, Université Henri Poincaré - Nancy I, Nancy, France.

[Jeanpierre and Charpillet, 2002] Jeanpierre, L. and Charpillet, F. (2002). Hidden markov models for medical diagnosis. In *Proc. HealtCom 2002*, pages 98–102, Nancy, France.

[Jelinek, 1997] Jelinek, F. (1997). *Statistical methods for speech recognition*. MIT Press.

[Jensen, 1996] Jensen, F. (1996). *An Introduction to Bayesian Networks*. UCL Press.

[Jensen et al., 1990] Jensen, F., Lauritzen, S., and Olesen, K. (1990). Bayesian updating in recursive graphical models by local computations. *Computational Statistical Quaterly*, 4 :269–282.

[Jordan, 1999] Jordan, M. I., editor (1999). *Learning in Graphical Models*. MIT Press.

[Jordan et al., 1999] Jordan, M. I., Ghahramani, Z., Jaakkola, T., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2) :183–233.

[Jr. and Young, 1999] Jr., E. S. and Young, J. (1999). Probabilistic temporal networks : A unified framework for reasoning with time and uncertainty. *Intl. Journal of Approximate Reasoning*, 20 :191–216.

[Kask et al., 2001] Kask, K., Dechter, R., Larrosa, J., and Cozman, F. (2001). Bucket-elimination for automated reasoning. Technical Report R92, UC Irvine ICS.

[Kiiveri et al., 1984] Kiiveri, H., Speed, T., and Carlin, J. (1984). Recursive causal models. *Journal of the Australian Mathematical Society*, 36 :30–52.

[Kim and Pearl, 1983] Kim, J. and Pearl, J. (1983). A computational model for combined causal and diagnostic reasoning in inference systems. In *Proceedings IJCAI-83*, pages 190–193, Karlsruhe, Germany.

[Kjaerulff, 1992] Kjaerulff, U. (1992). Optimal decomposition of probabilistic networks by simulated annealing. *Statistics and Computing*, 2 :7–17.

[Kjaerulff, 1998] Kjaerulff, U. (1998). *Nested junction trees*, pages 51–74. Kluwer Academic Publishers, Dordrecht, The Netherlands.

[Koenig and Simmons, 1996] Koenig, S. and Simmons, R. (1996). Unsupervised learning of probabilistic models for robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

[Koller and Pfeffer, 1997] Koller, D. and Pfeffer, A. (1997). Object-oriented bayesian networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-97)*, pages 302–313.

[Kong, 1986] Kong, A. (1986). *Multivariate Belief Functions and Graphical Models*. PhD thesis, Department of Statistics, Harvard University, Massachussetts.

[Kschischang et al., 2001] Kschischang, F., Frey, B., and Loeliger, H. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*.

[Lauritzen, 1982] Lauritzen, S. (1982). *Lectures on Contengency Tables*. University of Aalborg Press, Aalborg, Denmark, 2 edition.

[Lauritzen, 1988] Lauritzen, S. (1988). Local computation with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, 50 :157.

[Lauritzen, 1996] Lauritzen, S. (1996). *Graphical Models*. Clarendon, Oxford, UK.

[Lauritzen et al., 1990] Lauritzen, S., Dawid, A., Larsen, B., and Leimer, H. (1990). Independance properties of directed markov fields. *Networks*, 20 :491–505.

[Lauritzen and Wermuth, 1989] Lauritzen, S. and Wermuth, N. (1989). Graphical models for associations between variables, some of which are qualitative and some quantitative. *Annals of Statictics*, 17 :31–57.

[Lin, 1996] Lin, F. (1996). Embracing causality in specifying the indeterminate effects of actions. In *AAAI'96*, pages 670–676.

[Ling and Rudd, 1988] Ling, X. and Rudd, W. (1988). Combining opinions from several experts. *Applied Artificial Intelligence*, 3 :439–452.

[Llinas and Antony, 1993] Llinas, J. and Antony, R. (1993). Blackboard Concepts for Data Fusion Applications. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 7(2) :285–308.

[Lucas, 1998] Lucas, P. (1998). Analysis of notions of diagnosis. *Artificial Intelligence*, 105 :295–343.

[Luo and Kay, 1989] Luo, R. and Kay, M. (1989). Multisensor integration and fusion in intelligent systems. *IEEE Trans. on Systems, Man, and Cybernetics*, 19(5) :901–931.

[Madsen and Jensen, 1998] Madsen, A. and Jensen, F. (1998). Lazy propagation in junction trees. In Cooper, G. and Moral, S., editors, *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, pages 362–369, San Francisco, California. Morgan Kaufman.

[Martin and Moravec, 1996] Martin, M. and Moravec, H. (1996). Robot evidence grids. Technical Report CMU-RI-TR-96-06, Carnegie Mellon University, The Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania 15213.

[Maybeck, 1990] Maybeck, P. (1990). The kalman filter : An introduction to concepts. pages 194–204, New York, NY, USA. Springer-Verlag.

[McMichael et al., 1996] McMichael, D., Halgamuge, S., Hamlyn, G., Karan, M., and Okello, N. (1996). *Multisensor Data Fusion*. Lecture Notes.

[Meek et al., 2002] Meek, C., Chickering, D., and Heckerman, D. (2002). Autoregressive tree models for time-series analysis. In *Proceedings of the Second International SIAM Conference on Data Mining*, pages 229–244, Arlington, VA.

[Moravec, 1987] Moravec, H. (1987). Sensor fusion in certainty grids for mobile robots. pages 253–276.

[Murphy, 1999] Murphy, K. (1999). Filtering, smoothing and the junction tree algorithm. Technical report, University of Berkeley.

[Murphy, 2002] Murphy, K. (2002). *Dynamic Bayesian Networks : Representation, Inference and Learning*. PhD thesis, University of California, Berkeley.

[Murphy and Paskin, 2001] Murphy, K. and Paskin, M. (2001). Linear time inference in hierarchical hmms. In *Proceedings of the NIPS'01 Conference*.

[Naumann, 1998] Naumann, F. (1998). Data fusion and data quality. In *In Proc. of the New Techniques and Technologies for Statistics Seminar (NTTS)*, Sorrento, Italy.

[Olmsted, 1983] Olmsted, S. (1983). *On Representing and Solving Decision Problems*. PhD thesis, Department of Engineering-Economic Systems, Standford University, Stanford, California.

[Organization, 1997] Organization, W. H. (1997). Technical report, http ://www.who.int.

[Parsing, 1999] Parsing, S. (1999). Computational linguistics. pages 573–605.

[Pearl, 1982] Pearl, J. (1982). Reverand bayes on inference engines : A distributed hierarchical approach. In *Proceedings of the AAAI National Conference on AI*, pages 133–136, Pittsburgh.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic reasoning in intelligent systems : Networks of plausible inference*. Morgan Kaufman Publishers, Inc., San Mateo,CA, 2nd edition.

[Pearl, 2001] Pearl, J. (2001). *Causality - Models, reasoning and inference*. Cambrige University Press.

[Poole, 1994] Poole, D. (1994). Representing diagnosis knowledge. *Ann. Math. Artificial Intelligence*, 11 :33–50.

[Poole et al., 1987] Poole, D., Goebel, R., and Aleliunas, R. (1987). *Theoris : a logical reasoning system for defaults and diagnosis*. Springer, Berlin.

[Rabiner, 1989] Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–285.

[Rao, 1991] Rao, B. (1991). *Data Fusion Methods in Decentralized Sensing Systems*. PhD thesis, Dept. of Engineering Sciences, Oxford University.

[Shachter et al., 1994] Shachter, R., Andersen, S., and Szolovits, P. (1994). *Global conditionning for probabilistic inference in belief networks*, pages 514–524. Morgan Kaufman, San Francisco.

[Shafer, 1976] Shafer, G. (1976). *A mathematical theory of evidence*. Princeton University Press, Princeton, N.J.

[Shenoy, 1997] Shenoy, P. (1997). Binary join trees for computing marginals in the shenoy-shafer architecture. *International Journal of Approximate Reasoning*, 17 :239–263.

[Smets, 1988] Smets, P. (1988). *Belief functions. Non-Standard Logics for Automated Reasoning*. Academic Press, San Diego.

[Smyth et al., 1996] Smyth, P., Heckerman, D., and Jordan, M. (1996). Probabilistic Independance Networks for Hidden Markov Probability Models. Technical Report MSR-TR-96-03, Microsoft Research.

[Thomesse et al., 2002] Thomesse, J., Bellot, D., Boyer, A., Campo, E., Chan, M., Charpillet, F., Esteve, D., Fayn, J., Leschi, C., Noury, N., Rialle, V., Romary, L., Rubel, P., and Steenkeste, F. (2002). TIISSAD, Technologies de l'Information Intégrées aux Services de Soins à Domicile. Technical Report Décision d'aide 99 B0611 à 616, LORIA. Compte rendu de fin de recherche.

[Thomesse et al., 2001] Thomesse, J., Bellot, D., Boyer, A., Campo, E., Chan, M., Charpillet, F., Fayn, J., Leschi, C., Noury, N., Rialle, V., Romary, L., Rubel, P., Selmaoui, N., Steenkeste, F., and Virone, G. (2001). Integrated Information Technologies for patients remote follow-up and homecare. In *HealthCom 2001*.

[Thrun and Bücken, 1996] Thrun, S. and Bücken, A. (1996). Integrating grid-based and topological maps for mobile robot navigation. In AAAI, editor, *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Portland, Oregon. AAAI, AAAI.

[Thrun et al., 1998] Thrun, S., Gutmann, J., Fox, D., Burgard, W., and Kuipers, B. (1998). Integrating topological and metric maps for mobile robot navigation : A statistical approach. In *Proceedings of AAAI-98*. AAAI. http ://www.cs.cmu.edu/ thrun/papers/full.html.

[Verma and Pearl, 1988] Verma, T. and Pearl, J. (1988). Causal networks : Semantics and expressiveness. In *Proceedings of the 4th Workshop on Uncertainty in Artificial Intelligence*, pages 352–359, Mountain View, CA.

[Viterbi, 1967] Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. on Information Theory*, pages 260–269.

[Waltz and Llinas, 1990] Waltz, E. and Llinas, J. (1990). *Multisensor Data Fusion*. Boston-London.

[Wand and Wang, 1996] Wand, Y. and Wang, R. (1996). Anchoring data quality dimensions in ontological foundations. *Communications of the ACM*.

[Wang et al., 1996] Wang, R., Strong, D., and Guarascio, L. (1996). Beyond accuracy : What data quality means to data consumers.

[Wermuth and Lauritzebn, 1983] Wermuth, N. and Lauritzebn, S. (1983). Graphical and recursive models for contingency tables. *Biometrika*, 70 :37–52.

[Wu et al., 2001] Wu, X., Lucas, P., Kerr, S., and Dijkhuizen, R. (2001). Learning bayesian-network topologies in realistic medical domains. pages 302–308.

[Xu et al., 1992] Xu, L., Krzyzak, A., and Suen, C. (1992). Methods of combining multiple classifiers and their application to handwriting recognition. *IEEE Trans. on Systems, Man, and Cybernetics*, 22(3) :418–435.

[Yamauchi, 1995] Yamauchi, B. (1995). *Exploration and spatial learning in dynamic environments*. PhD thesis, Case Western Reserve University, Department of Computer Engineering and Science.

[Yannakakis, 1981] Yannakakis, M. (1981). Computing the minimum fill-in is np-complete. *SIAM Journal on Algebraic and Discrete Methods*, 2 :77–79.

[Zilberstein, 1995] Zilberstein, S. (1995). Models of Bounded Rationality : a concept paper. In *AAAI Fall Symposium on Rational Agency*, Cambridge, Massachusetts.

[Zweig and Russell, 1997] Zweig, G. and Russell, S. (1997). Compositional modeling with DPNs. Technical Report CSD-97-970.