Décomposition par paire pour l'optimisation combinatoire dans les modèles graphiques

Aurélie Favier, Simon de Givry, Andrés Legarra, Thomas Schiex
INRA Toulouse, France
{afavier,degivry,andres.legarra,tschiex}@toulouse.inra.fr

Abstract

Nous proposons une nouvelle décomposition additive des tables de probabilités qui préserve l'équivalence de la distribution jointe permettant de réduire la taille des potentiels, sans ajout de nouvelles variables. Nous formulons le problème de Most Probable Explanation (MPE) dans les réseaux probabilistes comme un problème de satisfaction de contraintes pondérées (Weighted Constraint Satisfaction Problem WCSP). Notre décomposition par paire permet de remplacer une fonction de coûts par des fonctions d'arités plus petites. Le WCSP résultant de cette décomposition est plus facile à résoudre par les techniques de l'état de l'art des WCSP. Même si tester la décomposition par paire est équivalent à tester l'indépendance de paire du réseau de croyances original, nous montrons comment le tester efficacement et l'appliquer, même avec des contraintes dures. De plus, nous inférons une information supplémentaire à partir des fonctions de coûts non binaires résultantes par projection&soustraction dans leurs fonctions binaires. Nous observons d'importantes améliorations grâce au pré-traitement avec la décompostion de paire et la projection&soustraction comparée aux solveurs actuels de l'état de l'art sur deux ensembles de problèmes difficiles.

1 Introduction

Les *modèles graphiques probabilistes* (Probabilistic Graphical Models PGM) permettent une approche générale pour les raisonnements automatiques sous incertitudes [6]. Les PGM couvrent les réseaux Bayésiens, les réseaux de Markov, mais également les formalismes déterministes comme les réseaux de contraintes. Ce papier s'intéresse au PGM discrets et à l'optimisation du problème MPE, qui consiste à maximiser le produit de fonctions positives sur un ensemble de variables discrètes. Dans la section 2, nous montrons comment, en utilisant une log-transformation, le problème MPE peut se reformuler en un WCSP, un formalisme général pour l'optimisation de contraintes qui consiste à minimiser la somme de *fonctions de coûts* positives sur un ensemble de variables dicrètes [12].

Les algorithmes exacts pour MPE (ou WCSP) sont pour la plupart basés sur l'algorithme de recherche Depth-First Branch and Bound (DFBB) ou sur des méthodes d'inférence, incluant l'élimination de variables, la jointure d'arbre et la compilation. Les méthodes d'inférence ont des problèmes de mémoire en général pour les problèmes importants et complexes (avec une largeur d'arbre importante). Les solveurs de l'état de l'art pour MPE combinent DFBB et les méthodes d'inférence à mémoire bornée (voir par exemple [13, 10]). DFBB est une méthode de recherche arborescente complète utilisant un espace mémoire linéaire. Pendant la recherche, pour les problèmes de minimisation, l'algorithme maintient un borne supérieure UB de la solution de coût minimum, en prenant le coût de la meilleure solution trouvée jusqu'alors. De plus, chaque nœud dans l'arbre de recherche est associé à une borne inférieure LB, une sous-estimation du coût minimum de la solution du sous-problème induit par le nœud courant. Si $LB \geq UB$, DFBB coupe l'espace de recherche sous le nœud courant.

Parmi les différentes techniques utilisées pour produire une bonne borne inférieure, les cohérences locales souples ont été introduites dans les WCSP. Leur complexité en temps dans le pire cas est exponentielle par rapport à la plus grande arité (nombre de variables) des fonctions de coûts [1], elles sont généralement seulement appliquées sur les fonctions de coûts de petites arités (2 ou 3). Par conséquent il est souhai-

Ce travail a bénéficié partiellement d'une aide de l'Agence Nationale de la Recherche portant la référence ANR-10-BLA-0214.

table de décomposer les fonctions de coûts en fonctions de coûts de plus petite arité. Notons que l'élimination de variables peut également bénéficier de cette décomposition (car le nombre de voisins d'une variable peut décroître).

Il est bien connu que les *indépendances conditionnelles* (IC) permettent de factoriser les distributions de probabilités. Toutefois ni les réseaux Baysiens ni les réseaux de Markov peuvent explicitement représenter toutes les IC parfaitement [6][1].

IC dépendants du contexte, telles que $(X \perp\!\!\!\perp Y \mid Z = z)$, qui ne sont pas explicites dans le réseau et qui peut être exploitée après avoir l'observation $Z = z$ ou réalisé l'élimination de variables ou éliminé des valeurs par cohérence locale souple. Dans les réseaux de Markov, plusieurs factorisations existent dans le cas des distributions non strictement positives. Dans la section 3, nous montrons comment exploiter efficacement l'indépendance de paire dans les WCSP pour améliorer DFBB avec les cohérences locales souples et l'élimination de variables.

2 Préliminaires

Un *Modèle Graphique Probabiliste* (PGM) (ou *distribution de Gibbs*) est défini par un produit de fonctions positives $\mathcal{F}$, sur un ensemble de variables discrètes $\mathcal{X}$, exprimant une information probabiliste ou déterministe [6]. Les sous-ensembles de $\mathcal{X}$ seront notés par des lettres en gras comme $\mathbf{S}$.

Définition 1 (PGM)**.** *Un PGM est un triplet $(\mathcal{X}, \mathcal{D}, \mathcal{F})$ avec $\mathcal{X} = \{X_1, \ldots, X_n\}$, un ensemble de variables, $\mathcal{D} = \{D_{X_1}, \ldots, D_{X_n}\}$, un ensemble de domaines finis de taille maximale $d = \max_{i=1}^{n} d_{X_i}$ ($d_{X_i} = |D_{X_i}|$), et $\mathcal{F} = \{f_1, \ldots, f_e\}$, un ensemble de fonctions à valeurs réelles positives, chacune est définie sur un sous-ensemble de variables $\mathbf{S}_i \subseteq \mathcal{X}$ (i.e., la portée). La distribution jointe est définie par :*

$$\mathbb{P}(\mathcal{X}) = \frac{\prod_{i=1}^{e} f_i(\mathbf{S}_i)}{\sum_{\mathcal{X}} \prod_{i=1}^{e} f_i(\mathbf{S}_i)}$$

Les réseaux Bayésiens représentent un cas particulier utilisant une probabilité conditionnelle par variable ($e = n$) et la constante de normalisation (dénominateur de $\mathbb{P}(\mathcal{X})$) vaut 1. Le problème *Most Probable Explanation* (MPE) consiste à trouver l'affectation la plus probable sur toutes les variables de $\mathcal{X}$ maximisant $\mathbb{P}(\mathcal{X})$.

Un *Problème de Satisfaction de Contraintes* (CSP) est un triplet $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ avec $\mathcal{X}$,$\mathcal{D}$ définis comme pour les PGM, et $\mathcal{C}$, un ensemble de contraintes. Chaque contrainte est définie comme une fonction booléenne sur un sous ensemble de variables $\mathbf{S}_i \subseteq \mathcal{X}$ indiquant les tuples dans $D_{\mathbf{S}_i} = \prod_{X \in \mathbf{S}_i} D_X$ sont autorisées ou non. Le problème consiste à trouver une *affectation possible* de $\mathcal{X}$, satisfaisant toutes les contraintes. Un *Problème de Satisfaction de Contraintes Pondérées* (WCSP) est une extension des CSP pour l'optimisation.

Définition 2 (WCSP)**.** *Un WCSP est un triplet $(\mathcal{X}, \mathcal{D}, \mathcal{W})$ avec $\mathcal{X}, \mathcal{D}$ définis comme pour les PGM et $\mathcal{W} = \{w_1, \ldots, w_e\}$, un ensemble de* fonctions de coûts*. Chaque fonction de coûts est définie sur un sous-ensemble de variables et prend des valeurs entières positives dans $E^+ = \mathbb{N} \cup \{\top\}$ ($\top = +\infty$ pouvant être associé aux affectations interdites[2]). Le but est de trouver une affectation valide de $\mathcal{X}$ minimisant $\sum_{i=1}^{e} w_i(\mathbf{S}_i)$.*

Un PGM $(\mathcal{X}, \mathcal{D}, \mathcal{F})$ peut être traduit en un WCSP $(\mathcal{X}, \mathcal{D}, \mathcal{W})$ avec $\forall i \in [1, e], w_i(\mathbf{S}_i) = \lceil -M \log(f_i(\mathbf{S}_i)) + C \rceil$ (M, C deux constantes pour la précision et la positivité des w_i lors de la transformation des réels en entiers) , il préserve l'ensemble des solutions optimales si une valeur suffisamment grande est utilisée pour M.

Dans la suite du papier, nous utilisons f pour représenter les fonctions de coûts. Etant donnée une affectation $t \in D_{\mathcal{X}}$, $t[\mathbf{S}]$ représente l'affectation t sur les variables de $\mathbf{S}$. Soit $f = f_1 + f_2$ la *somme (jointure) de deux fonctions de coûts* définie par $f(t) = f_1(t[\mathbf{S}_1]) + f_2(t[\mathbf{S}_2]), \forall t \in D_{\mathbf{S}_1 \cup \mathbf{S}_2}$. Soit $f = f_1 - f_2$ la *soustraction de deux fonctions de coûts* telle que $\mathbf{S}_2 \subseteq \mathbf{S}_1$ et $f(t) = f_1(t) - f_2(t[\mathbf{S}_2]), \forall t \in D_{\mathbf{S}_1}$ avec $\top - \top = \top$. $f[\mathbf{S}']$ représente *la projection d'une fonction de coûts* sur le sous-ensemble $\mathbf{S}'$ des variables de $\mathbf{S}$ telle que $\mathbf{S}' \subseteq \mathbf{S}$ et $\forall t' \in D_{\mathbf{S}'} f[\mathbf{S}'](t') = \min_{t \in D_{\mathbf{S}} \text{ s.t. } t[\mathbf{S}'] = t'} f(t)$. Une fonction de coûts sur deux (resp. une) variables est appelée une fonction de coûts *binaire* (resp. *unaire*). Une *fonction de coûts vide* a tous ses coûts égaux à zéro et est supprimée du WCSP.

L'algorithme de séparation évaluation (Depth-First Branch and Bound DFBB) est une méthode complète de recherche arborescente pour résoudre les WCSP. Les méthodes de cohérences locales souples produisent de fortes bornes inférieures pour DFBB par application de *Transformations Préservant l'Equivalence* (Equivalence Preserving Transformation EPT) [1]. Une *arc EPT* ajoute la projection $f[X]$ d'une fonction de coûts à une fonction de coûts unaire $f_1(X), X \in \mathbf{S}$ et soustrait $f[X]$ à f (*i.e.* remplace f par $f - f[X]$[3]) afin de garder un problème équivalent. L'application de la *cohérence d'arc souple* consiste à utiliser les arc

[1] Un exemple connu *le réseau de ségrégation* [3], où $\mathbb{P}(G_{i,jp} \mid G_{a,jp}, G_{a,jm}, S_{i,jp}) = 2\mathbb{P}(G_{i,jp} \mid G_{a,jp}, S_{i,jp})\mathbb{P}(G_{i,jp} \mid G_{a,jm}, S_{i,jp})$ car $(G_{a,jp} \perp\!\!\!\perp G_{a,jm} \mid G_{i,jp}, S_{i,jp})$, mais on ne peut pas le représenter comment un réseau bayésien.

[2] Par simplicité, nous utilisons $\top = +\infty$, mais nos résultats restent valides pour un $\top$ fini.

[3] $f - f[X]$ est bien une fonction de coûts toujours positive

EPT pour toutes les fonctions de coûts et toutes les directions ($\forall f(\mathbf{S}) \in \mathcal{W}, \forall X \in \mathbf{S}$) jusqu'à ce que toutes les projections soient vides. Une *arc EPT directionnelle* ajoute une projection $(f + f_2)[X]$ d'une fonction de coût binaire $f(X,Y)$ et unaire $f_2(Y)$ vers $f_1(X)$ et soustrait le résultat de la projection. L'application de la *cohérence d'arc souple directionnelle* (DAC) consiste à utiliser les arc EPT directionnelles sur toutes les fonctions de coûts suivant une seule direction ($X < Y$), indiquée par un ordre total sur $\mathcal{X}$, ainsi la terminaison est assurée. DFBB peut être également amélioré en utilisant l'*élimination de variables* à la volée [7]. Soient X une variable, $F = \{f_i \in \mathcal{W} \text{ s.t. } X \in \mathbf{S}_i\}$ et $\mathbf{S} = \bigcup_{f_i \in F} \mathbf{S}_i$. L'élimination de la variable X consiste à remplacer X et F par la projection $(\sum_{f \in F} f)[\mathbf{S} \setminus \{X\}]$ dans le WCSP courant. L'élimination de variables est normalement exponentielle en temps et en espace par rapport à la taille de $\mathbf{S}$ et est peu coûteuse si $|\mathbf{S}|$ est petite ou si X est affectée ou s'il est connecté au reste du problème par une seule fonction de coûts ou à travers au moins une contrainte bijective (contrainte d'égalité par exemple). DFBB appliquant l'élimination de variables i-bornée (toutes les variables avec $|\mathbf{S}| \leqslant i$ sont éliminées) et les cohérences locales souples (EDAC[4] pour les fonction de coûts binaires et ternaires [13]) est noté DFBB-VE(i) dans les résultats expérimentaux.

3 Décomposition de paire d'une fonction de coûts

Nous définissons tout d'abord la notion de *décomposition par paire*.

Définition 3. *Une* décomposition par paire *d'une fonction de coûts $f(\mathbf{S})$ par rapport à deux variables $X, Y \in \mathbf{S}$ ($X \neq Y$), est une réecriture de f en une somme de deux fonctions (positives) $f_1(\mathbf{S} \setminus \{Y\})$ et $f_2(\mathbf{S} \setminus \{X\})$ telles que :*

$$f(\mathbf{S}) = f_1(\mathbf{S} \setminus \{Y\}) + f_2(\mathbf{S} \setminus \{X\})$$

Un exemple de fonction décomposable est donnée par la Figure 1. Une décomposition par paire remplace une fonction de coûts d'artié $r = |\mathbf{S}|$ avec des fonctions de coûts d'arité plus petites $(r-1)$. Ce processus de décomposition peut être répété récursivement sur les fonctions résultantes, jusqu'à ce qu'aucune décomposition par paire soit trouvée pour chaque fonction de coûts restante. Une fonction de coûts $f(\mathbf{S})$ qui ne peut être décomposée par paire pour aucun choix de $X, Y \in \mathbf{S}$ est dite *non décomposable*. Un WCSP est *décomposé par paire* si toutes ses fonctions de coûts sont non décomposables. Cette propriété est respectée en appliquant le processus itératif précédemment décrit pour toutes les fonctions de coûts.

[4]EDAC combine AC et DAC. De plus elle teste pour chaque variable DAC avec un ordre mettant cette variable en premier sur le sous-problème constitué de $F = \{f_i \in W_{\mathbf{S}} \text{ t.q. } x \in \mathbf{S}_i$

W	X	Y	U	f
1	1	1	1	5
1	1	1	2	$\top$
1	1	2	1	$\top$
1	1	2	2	3
1	2	1	1	6
1	2	1	2	5
1	2	2	1	$\top$
1	2	2	2	$\top$
2	1	1	1	4
2	1	1	2	$\top$
2	1	2	1	$\top$
2	1	2	2	2
2	2	1	1	2
2	2	1	2	1
2	2	2	1	3
2	2	2	2	1

$=$

W	X	Y	f_1
1	1	1	5
1	1	2	3
1	2	1	5
1	2	2	$\top$
2	1	1	4
2	1	2	2
2	2	1	1
2	2	2	1

$+$

X	Y	U	f_2
1	1	1	0
1	1	2	$\top$
1	2	1	$\top$
1	2	2	0
2	1	1	1
2	1	2	0
2	2	1	2
2	2	2	0

FIG. 1 – $f(W,X,Y,U)$ fonction sur quatre variables Booléennes est décomposable par rapport à W,U comme $f_1(W,X,Y) + f_2(X,Y,U)$.

Le théorème suivant montre que tester si une fonction de coûts peut être décomposée par paire par rapport à X,Y est équivalent à tester l'indépendance de paire entre X et Y dans une distribution de probabilité appropriée.

Théorème 1 (Equivalence entre l'indépendance de paire et décomposition)**.** *Soit $\mathbf{S} = \{X,Y\} \cup \mathbf{Z}$ un ensemble de variables aléatoires, $f(\mathbf{S})$ une fonction de coûts sur $\mathbf{S}$ avec au moins une affectation autorisée, et une distribution $\mathbb{P}_f = \frac{1}{\sum_{\mathbf{S}} \exp(-f(\mathbf{S}))} \exp(-f(\mathbf{S}))$ (une* distribution de Gibbs *paramétrisée par f). $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ représente que X et Y sont indépendants par paire étant données toutes les autres variables de $\mathbb{P}_f$. Alors,*

$$(X \perp\!\!\!\perp Y \mid \mathbf{Z}) \iff f(X,\mathbf{Z},Y) = f_1(X,\mathbf{Z}) + f_2(\mathbf{Z},Y)$$

Au lieu de tester l'indépendance de paire dans $\mathbb{P}_f$, qui nécessite des sommations et des multiplications des nombres réels, nous proposons un test plus simple pour la décomposabilité de paire basée sur des égalités de différences de coûts. Pour une fonction de coûts décomposable par paire $f(X, \mathbf{Z}, Y) = f_1(X, \mathbf{Z}) + f_2(\mathbf{Z}, Y)$ n'ayant que des coûts finis et pour tout tuple $z \in D_{\mathbf{Z}}$, si nous considérons toutes les paires de valeurs k, l pour Y alors la différence $f(x, z, k) - f(x, z, l) = (f_1(x, z) + f_2(z, k)) - (f_1(x, z) + f_2(z, l)) = (f_2(z, k) - f_2(z, l))$ ne dépend pas de x. Lorsque $f(X, \mathbf{Z}, Y)$ n'est pas limité à des coûts finis, la soustraction d'un coût infini est mal définie. Par exemple, si $f_2(z, k)$ et $f_2(z, l)$ sont tous les deux égaux à $\top$ alors $f(x, z, k) = f(x, z, l) = \top$ pour tout x. Donc $f_1(x, z)$ peut prendre *n'importe quelle* valeur et la décomposition est toujours valable. Les coûts infinis offrent une liberté supplémentaire dans la décomposition et doivent être traités spécifiquement.

Pour avoir un test qui peut exploiter les coûts infinis, nous introduisons une extension de la structure de valuation des coûts $E = \mathbb{Z} \cup \{-\top, \top, \Omega\}$ incluant les coûts négatifs et un élément spécial asborbant Ω qui capture la liberté générée par la soustraction des coûts infinis. Soit $a \boxminus b$ représentant la différence de deux éléments $a, b \in E : a \boxminus b = a - b$, sauf pour $\top \boxminus \top = -\top \boxminus -\top = a \boxminus \Omega = \Omega \boxminus b = \Omega$, $\top \boxminus -\top = \top \boxminus c = c \boxminus -\top = \top$, $-\top \boxminus \top = -\top \boxminus c = c \boxminus \top = -\top$ avec $c \in \mathbb{Z}$. La comparaison de deux éléments $a, b \in E$ noté par $a \triangleq b$ est vraie si et seulement si (1) $a, b \in \mathbb{Z} \cup \{\top, -\top\}$ et $a = b$, ou (2) $a = \Omega$ ou $b = \Omega$. Nous avons alors :

Théorème 2. *Une fonction de coûts $f(X, \mathbf{Z}, Y)$ est décomposable par paire par rapport à X, Y ssi $\forall z \in D_{\mathbf{Z}}, \forall k, l \in D_Y (k < l)$[5], $\forall u, v \in D_{\mathbf{X}}$:*

$$f(u, z, k) \boxminus f(u, z, l) \triangleq f(v, z, k) \boxminus f(v, z, l)$$

Remarque 1. *Puisque $f(u, z, k) \boxminus f(u, z, l) = \Omega \triangleq a$, $\forall a \in E$, il suffit de trouver une valeur $u \in D_{\mathbf{X}}$ telle que $f(u, z, k) \boxminus f(u, z, l) \neq \Omega$ et de tester $\forall v \in D_{\mathbf{X}}$ $(v \neq u)$ telle que $f(v, z, k) \boxminus f(v, z, l) \neq \Omega$*
$f(u, z, k) \boxminus f(u, z, l) = f(v, z, k) \boxminus f(v, z, l)$
Puisque l'égalité est transitive le théorème 2 est bien vérifié.

Exemple 1. *Dans la Figure 1, nous avons $f(W, X, Y, U)$ décomposable par paire par rapport à W et U car :*

$$\begin{array}{ccccccc} f(1111) & \boxminus & f(1112) & \triangleq & f(2111) & \boxminus & f(2112) \\ 5 & \boxminus & \top & \triangleq & 4 & \boxminus & \top \\ f(1121) & \boxminus & f(1122) & \triangleq & f(2121) & \boxminus & f(2122) \\ \top & \boxminus & 3 & \triangleq & \top & \boxminus & 2 \\ f(1211) & \boxminus & f(1212) & \triangleq & f(2211) & \boxminus & f(2212) \\ 6 & \boxminus & 5 & \triangleq & 2 & \boxminus & 1 \\ f(1221) & \boxminus & f(1222) & \triangleq & f(2221) & \boxminus & f(2222) \\ \top & \boxminus & \top & \triangleq & 3 & \boxminus & 1 \end{array}$$

Ensuite, nous montrons comment contruire f_1, f_2 d'une fonction de coûts décomposable par paire en utilisant les projections et les soustractions des fonctions de coûts (voir la Figure 1 pour cette application).

Théorème 3. *Soit $f(X, \mathbf{Z}, Y)$ décomposable par paire par rapport à X, Y. Alors $f_1(X, \mathbf{Z}) = f[X, \mathbf{Z}]$ et $f_2(\mathbf{Z}, Y) = (f - f_1)[\mathbf{Z}, Y]$ est une décomposition valide de f, i.e. $f = f_1 + f_2$.*

Pour appliquer la décomposition par paire sur une fonction de coûts $f(\mathbf{S})$ avec $r = |\mathbf{S}|$ peut nécessiter $\frac{r(r-1)}{2}$ tests, *i.e.* vérifier pour toutes les paires de variables de $\mathbf{S}$. Et il peut produire des fonctions (non vides) d'arité (r-1). La répétition de la décomposition par paire pour les fonctions de coûts résultantes produira au plus $\min(\binom{r-p}{r}, 2^p)$ fonctions de coûts d'arité $(r - p)$ avec des portées différentes. Chaque test est linéaire par rapport à la taille des fonctions de coûts $(r - p)$-aire en $O(d^{(r-p)+1})$. Donc la complexité globale dans le pire des cas de l'application de la décomposition de paire pour e fonctions de coûts originales avec une arité r maximale est $O(e \sum_{p=0}^{r-1} 2^p (r - p)^2 d^{(r-p)+1}) = O(e \sum_{p=0}^{r-1} (r - p)^2 d^{r+1}) = O(e \frac{r(r+1)(2r+1)}{6} d^{r+1}) = O(er^3 d^{r+1})$ en temps et $O(e \max_{p=0}^{r-1} 2^p d^{(r-p)}) = O(ed^r)$ en espace (soit linéaire par rapport à la taille du problème).

A chaque étape, le choix des variables utilisées pour la décomposition et la manière de répartir les coûts dans f_1 et f_2 peuvent influencer la décomposabilité de f_1 et f_2 à l'itération suivante. Sous certaines conditions, il est possible de trouver une séquence de décomposition de paire (la paire de variables à sélectionner dans f et comment distribuer f dans f_1 et f_2) qui mène à une décomposition optimale (avec une arité minimale).

Propriété 1 (Décomposition minimale unique pour des fonctions de coûts finis [4])**.** *Une fonction de coûts avec seulement des coûts finis admet une décomposition* unique et minimale.

Ebauche de preuve. Si une fonction de coûts $f(\mathbf{S})$ a uniquement des coûts finis, alors sa distribution de probabilité associée $\mathbb{P}_f$ (voir Théorème 1) est positive. Le théorème d'*Hammersley & Clifford* [4] [6] dit qu'une

[5] Chaque élément de E admet un oppposé donc $-(f(u, z, k) \boxminus f(u, z, l)) = f(u, z, l) \boxminus f(u, z, k)$ d'où la condition de retester que dans un sens les égalités en prenant un ordre arbitraire sur les valeurs

[6] Voir aussi le Théorème 4.5 page 121 dans [6].

distribution positive se factorise en un unique réseau de Markov minimal $\mathcal{H}$. Le réseau est minimal dans le sens où la suppression d'une arête crée une nouvelle indépendance conditionnelle non présente dans $\mathbb{P}_f$. En prenant les cliques maximales dans $\mathcal{H}$, chaque clique donne la portée d'un facteur (une fonction positive) et $\mathbb{P}_f$ est égale au produit de ces facteurs divisé par une constante de normalisation. Cette factorisation est équivalente à une somme de fonctions de coûts en prenant $-\log(\mathbb{P}_f)$ (voir la preuve du Théorème 1). De cette décomposition résultante, il est facile de construire une séquence inverse de fonctions de coûts (positives) valides jusqu'à atteindre la fonction de coûts f originale. □

Cependant, dès qu'il y a des coûts infinis, le théorème précédent ne peut pas s'appliquer. Dans ce cas, nous proposons une approche heuristique dont le but est d'accumuler les coûts sur les premières variables dans l'ordre des variables de DAC. Cela devrait améliorer des bornes inférieures basées sur DAC. Nous testons donc des paires de variables dans $f(\mathbf{S})$ dans l'ordre DAC inverse[7] et projetons les fonctions de coûts (Théorème 3) sur la portée contenant les premières variables dans l'ordre DAC en premier. Dans la Figure 1, supposons l'ordre (W,X,Y,U), cela correspond à projeter en premier sur $\{W, X, Y\}$. Nous montrons que notre approche heuristique est capable de trouver une décomposition optimale pour quelques fonctions particulières.

Propriété 2 (Identification de structure d'arbre pour les contraitnes dures [11])**.** *Une fonction de coûts avec seulement des coûts infinis qui admet une décomposition en arbre de contraintes binaires est identifiable par notre stratégie de décomposition.*

Ce résultat vient de [11] et le fait qu'une décomposition par paire par rapport à X, Y supprimera une arête redondante $\{X, Y\}$ dans le réseau minimal[8] sans affecter l'ensemble des solutions. Par exemple, une contrainte globale d'égalité sur $\mathbf{S}$ sera décomposée en un arbre de contraintes d'égalité binaires : $f(\mathbf{S}) = \sum_{Y \in \mathbf{S} \setminus \{X\}} f_Y(X, Y)$ avec $f_Y(X, Y) \equiv (X = Y)$.

De la même façon, une fonction de coûts linéaire $f(X_1, \ldots, X_r) = \sum_{i=1}^{r} a_i X_i$ peut être décomposée en une somme de r fonctions de coûts unaires[9].

[7] Nous testons Y, U avant W, X si $\max(Y, U) > \max(W, X) \vee (\max(Y, U) = \max(W, X) \wedge \min(Y, U) > \min(W, X))$.

[8] Dans le cadre des CSP, le CSP binaire, appelé le *réseau minimal*, qui est la meilleure approximation d'une relation non binaire, est définie par les projections de la relation sur toutes les paires de variables [11].

[9] Notons que l'arc cohérence souple ferait de même si la fonction de coûts vide résultante est supprimée après les projections.

Travaux existants

Les travaux existants qui ont considéré la factorisation des fonctions de probabilités spécifiques comme les *noisy-or* et ses généralisations [5], ou des factorisations générales dédiées à l'inférence probabiliste [15]. Comparés à notre approche, ces travaux ajoutent de nouvelles variables. Une autre approche possible pour diminuer l'arité des fonctions de coûts est de passer à leur représentation duale [12]. Cela a été tester dans [2] pour Max-2SAT et était apparemment inefficace. Cela devrait probablement empirer pour de plus grands domaines.

4 Projection&Soustraction sur les fonctions de coûts binaires

Quand un WCSP est décomposé par paire, il est possible d'inférer une information supplémentaireà partir de la fonction de coûts résultante non binaire par projection & soustraction sur des fonctions de coûts de plus petites arités. Nous choisissons de projeter chaque fonction de coûts non binaire sur toutes les fonctions de coûts binaires possibles étant donnée la portée de la fonction en suivant un ordre des projections&soustractions compatible avec l'ordre des variables DAC (*i.e.* on projette d'abord sur les paires de variables avec les plus petites positions dans l'ordre DAC). Chaque projection $f_i(X, Y) = f[X, Y]$ est suivie par une soustraction $f = f - f_i$ afin de préserver l'équivalence du problème. Notons que f peut être vide après ces soustractions et est alors supprimée du problème. Notons aussi que la même fonction binaire peut recevoir des projections de coûts de plusieurs fonctions non binaires ayant des portées se chevauchant, résultant d'une plus forte inférence. La complexité dans le pire des cas de *projeter&soustraire* appliquée sur e fonctions de coûts avec r comme arité maximale est $O(er^2d^r)$ en temps et $O(er^2d^2)$ en espace.

Exemple 2. *Dans l'exemple de la Figure 1,* $f_1(W, X, Y) = b_1(W, X) + b_2(X, Y) + f_3(W, X, Y)$ *et* $f_2(X, Y, U) = b_3(X, U) + b_4(Y, U) + f_4(X, Y, U)$. *Finalement, une borne inférieure égale à 1 est déduite par arc cohérence souple appliquée sur* b_1.

W	X	b_1
1	1	3
1	2	5
2	1	2
2	2	1

X	Y	b_2
1	1	2
1	2	0
2	1	0
2	2	0

	i=2	i=3	i=4	i=5	i=6	i=7
Linkage (22)						
DFBB-VE(i)	14	14	15	13	12	10
DFBB-VE(i)+dec	16	**19**	17	13	14	13
DFBB-VE(i)+ps	**17**	16	17	18	16	16
DFBB-VE(i)+dec+ps	16	18	**19**	**19**	**19**	**17**
Grids (32)						
DFBB-VE(i)	28	28	25	24	17	18
DFBB-VE(i)+dec	29	29	29	28	28	31
DFBB-VE(i)+ps	28	28	28	23	25	24
DFBB-VE(i)+dec+ps	**31**	**30**	**31**	**31**	**32**	**32**

TAB. 1 – Nombre d'instances résolues en moins d'une heure par chaque méthode

X	U	b_3
1	1	0
1	2	0
2	1	1
2	2	0

Y	U	b_4
1	1	0
1	2	0
2	1	1
2	2	0

W	X	Y	f_3
1	1	1	0
1	1	2	0
1	2	1	0
1	2	2	$\top$
2	1	1	0
2	1	2	0
2	2	1	0
2	2	2	0

X	Y	U	f_4
1	1	1	0
1	1	2	$\top$
1	2	1	$\top$
1	2	2	0
2	1	1	0
2	1	2	0
2	2	1	0
2	2	2	0

5 Résultats expérimentaux

DFBB-VE(i)[10] maintenant EDAC et utilisant un ordre dynamique d'affectation des variables basé sur les conflits et une pondération des fonctions de coûts [9] a obtenu les meilleurs résultats de l'évaluation de MPE de UAI'08[11] (et UAI'10[12] pour le cas à 20 minutes) excepté pour deux benchmarks difficiles : les réseaux d'analyse de liaison et les réseaux en grille.

Les réseaux d'analyse de liaison génétique (Linkage). Les instances ont entre 334 et 1289 variables. Le domaine maximal de taille d est entre 3 et 7 et l'arité la plus grande est 5. Cette famille de benchmarks est constituée de 22 problèmes.

Les réseaux en grille (Grids). Chaque problème est une grille $l \times l$ et chaque ternaire CPT est générée aléatoirement et uniformément. 50, 75 or 90% des CPT sont déterministes et les variables sont Booléennes. Pour ces instances l varie de 12 à 50 (*i.e.* $n = 2500$ variables). Cette famille de benchmarks a 32 problèmes.

Les expérimentations[10] ont été réalisées sur un ordinateur 2.6 GHz Intel Xeon avec 4GB sous Linux 2.6. Les temps CPU totaux sont en secondes et limités à 1 heure pour chaque instance ("-" représente un dépassement de temps). Aucune borne supérieure n'est donnée au départ. Nous testons l'application de la décomposition par paire (DFBB-VE(i)+dec), projeter&soustraire les fonctions de coûts n-aires(DFBB-VE(i)+ps), et la combinaison des deux techniques (DFBB-VE(i)+dec+ps). La décomposition par paire, de projeter&soustraire et l'élimination de variables de degré $|\mathbf{S}| \leq i$ sont effectuées seulement en prétraitement. L'élimination à la volée des variables de degré au plus 2 est réalisé durant la recherche. Toutes les méthodes utilisent l'heuristique *MinFill* pour l'ordre d'élimination de variables et l'ordre inverse est utilisé pour DAC. La Table 1 résume le nombre de problèmes résolus par les différentes versions de DFBB-VE(i). La décomposition par paire et projeter&soustraire résolvent plus de problèmes que DFBB-VE(i) seul et leur combinaison obtient les meilleurs résultats, sauf pour les Linkage avec $i = 2, 3$. Remarquablement, toutes les instances de grilles sont résolues avec de l'élimination de variables bornée pour un i suffisamment grand. montrant l'effet de la décomposition par paire, spécialement quand il y a suffisament de déterminisme. Une factorisation similaire est observée dans [14] pour les CSP.

Nous avons ensuite comparé la meilleure méthode avec AND/OR Branch and Bound avec cache et minibucket statique j-borné (AOBB-C+SMB(j)) qui a eu les meilleurs résultats pour Linkage et Grids à l'évaluation UAI'08. Cette méthode est implémentée dans `aolibWCSP`[13]. La valeur de j est choisie comme dans [10]. La Table 2 donne les tailles des problèmes (n, d), la treewidth (w) et résume le temps CPU total utilisé pour les différentes méthodes avec le i correspondant (en choisissant $i \in [2, 7]$ permettant d'obtenir les

[10] `mulcyber.toulouse.inra.fr/projects/toulbar2` version 0.9.4.
[11] `graphmod.ics.uci.edu/uai08/Evaluation`
[12] `www.cs.huji.ac.il/project/UAI10`
[13] `graphmod.ics.uci.edu/group/aolibWCSP`

Problème				DFBB-VE(i)			DFBB-VE(i) +dec+ps					AOBB-C+SMB(j) +VE(i)		AOBB-C+SMB(j) +VE(i)+dec+ps	
	n	d	w	time (s)	i	n'	time (s)	i	n'	j	i	time (s)	n'	time (s)	n'
Linkage															
ped1	334	4	16	0.12	3	159	**0.07**	3	142	10	3	0.1	159	0.08	142
ped7	1068	4	38	4.04	2	375	**1.18**	4	213	20	4	1915.56	274	131.14	213
ped9	1118	4	31	-			**3.36**	6	147	20	6	-	179	104.62	147
ped18	1184	5	24	149.71	4	365	**3.19**	5	213	20	5	54.67	255	18.82	213
ped19	793	5	32	-			-			20	6	-	337	-	304
ped20	437	5	23	3.46	4	95	**0.39**	6	42	16	6	407.6	68	66.53	42
ped23	402	5	26	0.09	4	79	**0.05**	3	98	12	3	6.05	114	1.52	98
ped25	1289	5	29	1207.97	4	269	**0.65**	6	102	20	6	25.91	158	32.51	102
ped30	1289	5	24	543.20	2	633	**5.44**	5	213	20	5	28.39	272	10.10	231
ped33	798	4	30	0.84	3	272	**0.54**	6	151	18	6	21.33	175	8.35	151
ped34	1160	5	36	1.13	2	326	**0.36**	5	167	20	5	-	196	24.56	167
ped37	1032	4	22	0.21	5	103	**0.11**	4	120	10	4	87.18	141	9.58	120
ped38	724	5	18	0.24	5	122	**0.18**	5	97	12	5	137.60	122	124.70	97
ped39	1272	5	22	16.68	4	183	**0.24**	5	107	18	5	6.87	148	2.60	107
ped41	1062	5	35	-			**302.05**	4	308	20	4	-	372	1271.31	308
ped42	448	5	24	1.94	4	140	**0.34**	6	81	16	6	240.94	95	155.39	81
ped44	811	4	31	-			505.46	5	215	20	5	2631.71	248	**333.89**	215
ped50	514	6	18	0.90	4	133	**0.18**	4	118	12	4	316.92	133	521.92	118
Grids															
90-24-1	576	2	35	0.07	3	566	**0.04**	3	291	18	3	2323.01	566	0.63	291
90-26-1	676	2	41	0.29	3	668	**0.07**	3	500	16	3	2821.66	668	20.16	500
90-30-1	900	2	49	5.50	2	766	**0.26**	4	382	18	4	-	766	3.57	382
90-34-1	1156	2	63	328.42	2	1052	**0.48**	5	518	20	5	-	1037	14.41	518
90-38-1	1444	2	64	-			1.96	6	249	20	6	-	983	**0.64**	249

TAB. 2 – Comparaison avec/sans le pré-traitement dec+ps pour DFBB-VE(i) et AOBB-C+SMB(j)

meilleurs résultats pour chaque instance) et j. Le pré-traitement "dec+ps" améliore globalement les résultats de DFBB-VE(i) et AOBB-C+SMB(j). Le temps du pré-traitement était toujours inférieur à une seconde dans nos expérimentations (n', nombre de variables après le pré-traitement). DFBB-VE(i)+dec+ps a obtenu les meilleurs résultats pour tous les problèmes sauf pour ped44 and 90-38-1. Notons que la variation de i pour DFBB-VE(i)+dec+ps est plus petite que celle de j pour AOBB-C+SMB(j)[14].

6 Conclusion

La décomposition par paire combinée avec les projections sur les fonctions de coûts binaires et l'élimination de variables est une puissante technique à l'intérieur de DFBB pour MPE. Les travaux futurs devraient être fait sur la décomposition par paire approchée. D'autres expérimentations sur d'autres problèmes incluant l'inférence probabiliste devraient également être réalisées.

[14]Un bon moyen de régler la valeur de i est de prendre la valeur correspondant au maximum de la distribution des degrés pour le modèle graphique considéré (*i.e.* $i = 5$ pour Linkage et $i = 6$ pour Grids).

A Preuve du Théorème 1

Dans la suite, nous utilisons $\mathbb{P}$ comme raccourci pour $\mathbb{P}_f$ et $C = \sum_{\mathbf{S}} \exp(-f(\mathbf{S})) > 0$, une constante de normalisation. Par définition des indépendances conditionnelles [8], nous avons :
$(X \perp\!\!\!\perp Y \mid \mathbf{Z}) \iff \mathbb{P}(X, Y, \mathbf{Z}) = \mathbb{P}(X \mid \mathbf{Z})\mathbb{P}(Y \mid \mathbf{Z})\mathbb{P}(\mathbf{Z})$
$\iff \mathbb{P}(X, Y, \mathbf{Z}) = \frac{\mathbb{P}(X,\mathbf{Z})}{\mathbb{P}(\mathbf{Z})}\mathbb{P}(Y, \mathbf{Z})$, with $\mathbb{P}(\mathbf{Z}) > 0$.

$\boxed{\Longrightarrow}$
Supposons $\mathbb{P}(X, Y, \mathbf{Z}) = \mathbb{P}(X \mid \mathbf{Z})\ \mathbb{P}(Y \mid \mathbf{Z})\ \mathbb{P}(\mathbf{Z})$. Nous avons $f(X, \mathbf{Z}, Y) = -\log(C\ \mathbb{P}(X, Y, \mathbf{Z})) = -\log(C\ \mathbb{P}(X \mid \mathbf{Z})\ \mathbb{P}(Y \mid \mathbf{Z})\ \mathbb{P}(\mathbf{Z}))$. Définissons $f'_1(X, \mathbf{Z}) = -\log(\mathbb{P}(X \mid \mathbf{Z}))$ et $f'_2(\mathbf{Z}, Y) = -\log(\mathbb{P}(Y \mid \mathbf{Z})\mathbb{P}(\mathbf{Z}))$, deux fonctions de coût positives.
Ainsi $f(X, \mathbf{Z}, Y) = f'_1(X, \mathbf{Z}) + f'_2(\mathbf{Z}, Y) - \log(C)$. Car f est positive, nous avons $\forall x \in D_X, \forall y \in D_Y, \forall z \in D_{\mathbf{Z}}, f'_1(x, z) + f'_2(z, y) - \log(C) \geq 0$. Si $\log(C)$ est négatif, nous ajoutons $-\log(C)$ à f'_1 ou f'_2 afin d'obtenir f_1, f_2.
Sinon, pour chaque $z \in D_{\mathbf{Z}}$, nous décomposons $\log(C) = c_z^1 + c_z^2$ en deux nombres positifs.
Soit $\hat{x}_z = \text{argmin}_{x \in D_X} f'_1(x, z)$ et $\hat{y}_z = \text{argmin}_{y \in D_Y} f'_2(z, y)$. De plus, nous avons $f'_1(\hat{x}_z, z) + f'_2(z, \hat{y}_z) - \log(C) = f(\hat{x}_z, z, \hat{y}_z) \geq 0$. Une solution possible est $c_z^1 = \min(f'_1(\hat{x}_z, z), \log(C))$ et $c_z^2 = \log(C) - c_z^1$.

Either $c_z^1 = \log(C) \leq f_1'(\hat{x}_z, z)$ et $c_z^2 = 0$, ou $c_z^1 = f_1'(\hat{x}_z, z)$ et $c_z^2 = \log(C) - f_1'(\hat{x}_z, z)$.
Dans ce cas, $f_2'(z, \hat{y}_z) - c_z^2 = f_2'(z, \hat{y}_z) + f_1'(\hat{x}_z, z) - \log(C) \geq 0$, ainsi $\forall y \in D_Y, f_2'(z, y) - c_z^2 \geq 0$. Nous définissons $\forall x, y, z,\ f_1(x, z) = f_1'(x, z) - c_z^1$ et $f_2(z, y) = f_2'(z, y) - c_z^2$, qui sont des nombres positifs Finalement, nous en déduisons $f(X, \mathbf{Z}, Y) = f_1(X, \mathbf{Z}) + f_2(\mathbf{Z}, Y)$.

$\boxed{\Longleftarrow}$
Supposons que nous ayons trois fonctions de coûts $f(X, \mathbf{Z}, Y), f_1(X, \mathbf{Z}), f_2(\mathbf{Z}, Y)$ telles que $f(X, \mathbf{Z}, Y) = f_1(X, \mathbf{Z}) + f_2(\mathbf{Z}, Y)$.
Nous avons
$\exp(-f(X, \mathbf{Z}, Y)) = \exp(-f_1(X, \mathbf{Z})) \exp(-f_2(\mathbf{Z}, Y))$.
Par marginalisation, nous avons
$\mathbb{P}(X, \mathbf{Z}) = \sum_Y \mathbb{P}(X, Y, \mathbf{Z}) = \frac{1}{C} \sum_Y \exp(-f(X, \mathbf{Z}, Y))$
$= \frac{1}{C} \exp(-f_1(X, \mathbf{Z})(\sum_Y \exp(-f_2(\mathbf{Z}, Y)))$, $\mathbb{P}(Y, \mathbf{Z})$
$= \sum_X \mathbb{P}(X, Y, \mathbf{Z})$
$= \frac{1}{C} \exp(-f_2(\mathbf{Z}, Y))\ (\sum_X \exp(-f_1(X, \mathbf{Z})))$
et $\mathbb{P}(\mathbf{Z}) = \sum_{X,Y} \mathbb{P}(X, Y, \mathbf{Z})$
$= \frac{1}{C}(\sum_X \exp(-f_1(X, \mathbf{Z})))(\sum_Y \exp(-f_2(\mathbf{Z}, Y)))$.
Finalement, nous en déduisons $\frac{\mathbb{P}(X, \mathbf{Z})}{\mathbb{P}(\mathbf{Z})} \mathbb{P}(Y, \mathbf{Z}) = \frac{1}{C} \exp(-f_1(X, \mathbf{Z})) \exp(-f_2(\mathbf{Z}, Y)) = \mathbb{P}(X, Y, \mathbf{Z})$.

B Preuve du Théorème 2

Il est toujours possible de décomposer une fonctiond de coûts f en trois fonction de coûts (positives) $f(X, \mathbf{Z}, Y) = f_1(X, \mathbf{Z}) + f_2(\mathbf{Z}, Y) + \Delta(X, \mathbf{Z}, Y)$ (f_1, f_2 peuvent être égales à la fonction constante nulle). Une fonction de coût non nulle est appelée *réductible* si f_1 ou f_2 sont des fonctions de coûts non nulles. Sinon elle est appelée *irréductible*. Nous avons une condition spécifique pour tester la *réductibilité*.

Propriété 3. *Une fonction de coût non nulle* $f(X, \mathbf{Z}, Y)$ *est* réductible *si* $\exists x \in D_X, \exists z \in D_{\mathbf{Z}}$ *t.q.* $\min_{y \in D_Y} f(x, z, y) > 0$ *(i.e.* $f_1 = f[X, \mathbf{Z}] \neq 0$*) ou* $\exists y \in D_Y, \exists z \in D_{\mathbf{Z}}$ *t.q.* $\min_{x \in D_X} f(x, z, y) > 0$ *(i.e.* $f_2 = f[\mathbf{Z}, Y] \neq 0$*).*

Propriété 4. *Si* $f_1(X, \mathbf{Z})$, $f_2(\mathbf{Z}, Y)$, $\Delta(X, \mathbf{Z}, Y)$, *trois fonctions de coûts (positives), est une réduction maximale de* $f(X, \mathbf{Z}, Y)$ *alors* $\Delta(X, \mathbf{Z}, Y)$ *est irréductible.*

En utilisant la propriété 3, nous prouvons le lemme suivant.

Lemme 1. *Si* $f(X, \mathbf{Z}, Y)$ *une fonction de coûts non nulle est* irréductible *alors* $\exists z \in D_{\mathbf{Z}}, \exists k, l \in D_Y (k \neq l), \exists u, v \in D_X (u \neq v)$ *t.q.* $f(u, z, k) \boxminus f(u, z, l) \neq f(v, z, k) \boxminus f(v, z, l)$.

Preuve par l'absurde. Supposons $\forall z \in D_{\mathbf{Z}}, \forall k, l \in D_Y (k \neq l), \forall u, v \in D_X (u \neq v)$ tel que $f(u, z, k) \boxminus f(u, z, l) \triangleq f(v, z, k) \boxminus f(v, z, l)$. Nous rappelons que les fonctions de coûts ont leur coût dans $E^+ = \mathbb{N} \cup \{\top\}$ et non dans E. Nous avons :
Premier cas :
$\exists u, z, k, l$ **t.q.** $f(u, z, k) \boxminus f(u, z, l) \not\triangleq 0$
$\boxed{f(u, z, k) \boxminus f(u, z, l) > 0 \text{ ou } f(u, z, k) \boxminus f(u, z, l) = \top}$
Donc $0 \leqslant f(u, z, l) < f(u, z, k)$.
De plus $\forall v \in D_X$,
$f(u, z, k) \boxminus f(u, z, l) \triangleq f(v, z, k) \boxminus f(v, z, l)$.
Alors $\forall v \in D_X,\ f(v, z, k) \boxminus f(v, z, l) > 0$, ou $f(v, z, k) \boxminus f(v, z, l) = \top$, ou $f(v, z, k) \boxminus f(v, z, l) = \Omega$.
Par conséquent $0 \leqslant f(v, z, l) < f(v, z, k)$ ou $f(v, z, k) = f(v, z, l) = \top$. Donc $\min_{v \in D_X} f(v, z, k) > 0$.

$\boxed{f(u, z, k) \boxminus f(u, z, l) < 0 \text{ or } f(u, z, k) \boxminus f(u, z, l) = -\top}$
Donc $0 \leqslant f(u, z, k) < f(u, z, l)$.
De plus $\forall v \in D_X$,
$f(u, z, k) \boxminus f(u, z, l) \triangleq f(v, z, k) \boxminus f(v, z, l)$.
Ainsi $\forall v \in D_X,\ f(v, z, k) \boxminus f(v, z, l) < 0$, ou $f(v, z, k) \boxminus f(v, z, l) = -\top$, ou $f(v, z, k) \boxminus f(v, z, l) = \Omega$.
Par conséquent $0 \leqslant f(v, z, k) < f(v, z, l)$ ou $f(v, z, k) = f(v, z, l) = \top$. Donc $\min_{v \in D_X} f(v, z, l) > 0$.
Second cas :
$\forall u, z, k, l$ **t.q.** $f(u, z, k) \boxminus f(u, z, l) \triangleq 0$
$\boxed{f(u, z, k) \boxminus f(u, z, l) = \Omega}$
Donc $f(u, z, k) = f(u, z, l) = \top$.
De plus $\forall p \in D_Y,\ f(u, z, p) \boxminus f(u, z, l) = 0$ ou $f(u, z, p) \boxminus f(u, z, l) = f(u, z, p) \boxminus \top = \Omega$. Ainsi $f(u, z, p) = \top$. Donc $f(u, z, l) = f(u, z, k) = f(u, z, p) = \top$, et $\min_{p \in D_Y} f(u, z, p) = \top > 0$.

$\boxed{f(u, z, k) = f(u, z, l) > 0}$
Donc $f(u, z, k) \boxminus f(u, z, l) = 0$ et par hypothèse, $\forall p \in D_Y,\ f(u, z, p) \boxminus f(u, z, l) = 0$.
Ainsi $f(u, z, k) \boxminus f(u, z, l) = f(u, z, p) \boxminus f(u, z, l)$ et $0 < f(u, z, l) = f(u, z, k) = f(u, z, p)$.
Donc $\min_{p \in D_Y} f(u, z, p) > 0$.

$\boxed{f(u, z, k) = f(u, z, l) = 0}$
Par hypothèse, $\forall p \in D_Y,\ f(u, z, p) \boxminus f(u, z, l) \triangleq 0$.
Alors $f(u, z, p) \boxminus 0 \triangleq 0$. Ainsi $f(u, z, p) \boxminus f(u, z, l) \neq \Omega$ et donc $f(u, z, k) \boxminus f(u, z, l) = f(u, z, p) \boxminus f(u, z, l)$.
Finalement $\forall p \in D_Y, f(u, z, p) = 0$. Nous pouvons conclure en utilisant la propriété 3 que $f(X, \mathbf{Z}, Y)$ est réductible ou égal à la fonction nulle (si de manière récursive on est toujours dans le dernier sous-cas). $\square$

L'opération $a \boxplus b$ représente l'addition de deux éléments de $a, b \in E$: $a \boxplus b = a + b$ sauf

$\top \boxplus -\top = -\top \boxplus \top = a \boxplus \Omega = \Omega \boxplus b = \Omega$,
$\top \boxplus \top = \top \boxplus c = c \boxplus \top = \top$,
$-\top \boxplus -\top = -\top \boxplus c = c \boxplus -\top = -\top$ avec $c \in \mathbb{Z}$.

Maintenant nous pouvons prouver le théorème 2.

$\boxed{\Longrightarrow}$ Supposons $f(X, \mathbf{Z}, Y) = f_1(X, \mathbf{Z}) + f_2(\mathbf{Z}, Y)$ et $f_1(X, \mathbf{Z})$, $f_2(\mathbf{Z}, Y)$ fonctions positives ($0 \leqslant f_1(X, \mathbf{Z}) \leqslant f(X, \mathbf{Z}, Y)$). Alors les systèmes suivants $\mathcal{S}_z$ d'équations linéaires (utilisant $\boxplus$ et $\stackrel{\circ}{=}$ operateurs) ont une solution.

$$(\mathcal{S}_z) \begin{cases} f_1(1z) \boxplus f_2(z1) \stackrel{\circ}{=} f(1z1) \\ f_1(uz) \boxplus f_2(zk) \stackrel{\circ}{=} f(uzk) \quad \forall u, k \\ f_1(d_X z) \boxplus f_2(z d_Y) \stackrel{\circ}{=} f(d_X z d_Y) \end{cases}$$

$\forall z \in D_{\mathbf{Z}}$, $\forall k, l \in D_Y$, $\forall u \in D_X$, de $\mathcal{S}_z$, nous déduisons

$$\begin{aligned} f(uzk) \boxminus f(uzl) &\stackrel{\circ}{=} (f_1(uz) \boxplus f_2(zk)) \boxminus (f_1(uz) \boxplus f_2(zl)) \\ &\stackrel{\circ}{=} (f_1(uz) \boxminus f_1(uz)) \boxplus (f_2(zk) \boxminus f_2(zl)) \end{aligned}$$

1^{er} **cas :** $f_1(uz) \in \mathbb{N}$
$f(uzk) \boxminus f(uzl) \stackrel{\circ}{=} (f_1(uz) \boxminus f_1(uz)) \boxplus (f_2(zk) \boxminus f_2(zl)) \stackrel{\circ}{=} 0 \boxplus (f_2(zk) \boxminus f_2(zl)) \stackrel{\circ}{=} f_2(zk) \boxminus f_2(zl)$

2^{nd} **cas :** $f_1(uz) = \top$
$f(uzk) \boxminus f(uzl) \stackrel{\circ}{=} (f_1(uz) \boxminus f_1(uz)) \boxplus (f_2(zk) \boxminus f_2(zl)) \stackrel{\circ}{=} \Omega \boxplus (f_2(zk) \boxminus f_2(zl)) \stackrel{\circ}{=} \Omega$

Nous pouvons conclure que $\forall z \in D_{\mathbf{Z}}, \forall k, l \in D_Y$ $(k \neq l) \stackrel{\circ}{=}_{u \in D_X} f(uzk) \boxminus f(uzl)$

$\boxed{\Longleftarrow}$ Supposons $\forall z \in D_{\mathbf{Z}}, \forall k, l \in D_Y$ $(k < l), \forall u, v \in D_{\mathbf{X}}$: $f(uzk) \boxminus f(uzl) \stackrel{\circ}{=} f(vzk) \boxminus f(vzl)$.
De plus, nous avons $f(X, \mathbf{Z}, Y) = f_1(X, \mathbf{Z}) + f_2(\mathbf{Z}, Y) + \Delta(X, \mathbf{Z}, Y)$ $f_1(X, \mathbf{Z})$, $f_2(\mathbf{Z}, Y)$ et $\Delta(X, \mathbf{Z}, Y)$ des fonctions positives qui définissent les systèmes linéaires suivants :

$$(\mathcal{S}'_z) \begin{cases} f_1(1z) \boxplus f_2(z1) \boxplus \Delta(1z1) \stackrel{\circ}{=} f(1z1) \\ f_1(uz) \boxplus f_2(zk) \boxplus \Delta(uzk) \stackrel{\circ}{=} f(uzk) \quad \forall u, k \\ f_1(d_X z) \boxplus f_2(z d_Y) \boxplus \Delta(d_X z d_Y) \stackrel{\circ}{=} f(d_X z d_Y) \end{cases}$$

$$\begin{aligned} & f(uzk) \boxminus f(uzl) \\ & \stackrel{\circ}{=} (f_1(uz) \boxplus f_2(zk) \boxplus \Delta(uzk)) \boxminus (f_1(uz) \boxplus f_2(zl) \boxplus \Delta(uzl)) \\ & \stackrel{\circ}{=} f_1(uz) \boxplus f_2(zk) \boxplus \Delta(uzk) \boxplus (-f_1(uz)) \boxplus (-f_2(zl)) \\ & \boxplus (-\Delta(uzl)) \\ & \stackrel{\circ}{=} (f_1(uz) \boxminus f_1(uz)) \boxplus (f_2(zk) \boxminus f_2(zl)) \boxplus (\Delta(uzk) \boxminus \Delta(uzl)) \end{aligned}$$

Supposons que $f(X, \mathbf{Z}, Y)$ ne se décompose pas en $\{f_1(X, \mathbf{Z}), f_2(\mathbf{Z}, Y)\}$, avec la propriété 4 nous pouvons trouver $\Delta(X, \mathbf{Z}, Y)$ tel que $\Delta(X, \mathbf{Z}, Y)$ est une fonction de coût non nulle irréductible. Avec le théorème 1 nous avons $\exists z \in D_{\mathbf{Z}}$, $k, l \in D_Y, k < l$ et $u, v \in D_X, u < v$ t.q. $\Delta(uzk) \boxminus \Delta(uzl) \not\stackrel{\circ}{=} \Delta(vzk) \boxminus \Delta(vzl)$.

Nous en déduisons que $\Delta(uzk) \boxminus \Delta(uzl) \neq \Omega$ et $\Delta(vzk) \boxminus \Delta(vzl) \neq \Omega$ avec la définiton de $\stackrel{\circ}{=}$.
Nous avons également $f(uzk) \boxminus f(uzl) \neq \Omega \neq f(vzk) \boxminus f(vzl)$. Alors $f(uzk) \boxminus f(uzl) \stackrel{\circ}{=} f(vzk) \boxminus f(vzl) \stackrel{\circ}{=} \varphi$ t.q. $\varphi \in \mathbb{Z} \cup \{\top, -\top\}$ donc $f_1(uz) \boxminus f_1(uz) = 0 = f_1(vz) \boxminus f_1(vz)$ et $f_2(zk) \boxminus f_2(zl) \neq \Omega$

En utilisant les propriétés précédentes :

$$\begin{aligned} & \Delta(uzk) \boxminus \Delta(uzl) \neq \Delta(vzk) \boxminus \Delta(vzl) \\ & f_2(zk) \boxminus f_2(zl) \boxplus \Delta(uzk) \boxminus \Delta(uzl) \\ & \quad \neq f_2(zk) \boxminus f_2(zl) \boxplus \Delta(vzk) \boxminus \Delta(vzl) \\ & f_1(uz) \boxminus f_1(uz) \boxplus f_2(zk) \boxminus f_2(zl) \boxplus \Delta(uzk) \boxminus \Delta(uzl) \\ & \quad \neq f_1(vz) \boxminus f_1(vz) \boxplus f_2(zk) \boxminus f_2(zl) \boxplus \Delta(vzk) \boxminus \Delta(vzl) \\ & f_1(uz) \boxplus f_2(zk) \boxplus \Delta(uzk) \boxminus (f_1(uz) \boxplus f_2(zl) \boxplus \Delta(uzl)) \\ & \quad \neq f_1(vz) \boxplus f_2(zk) \boxplus \Delta(vzk) \boxminus (f_1(vz) \boxplus f_2(zl) \boxplus \Delta(vzl)) \\ & f(uzk) \boxminus f(uzl) \neq f(vzk) \boxminus f(vzl) \end{aligned}$$

C'est en contradiction avec la première hypothèse : $\forall\ k, l \in D_Y$, $\forall\ z \in D_{\mathbf{Z}}$ $f(uzk) \boxminus f(uzl) \stackrel{\circ}{=} f(vzk) \boxminus f(vzl)$.

C Preuve du Théorème 3

Lemme 2. *Soit une fonction de coûts $f(X, \mathbf{Z}, Y)$ décomposable par paire par rapport à X, Y.*
Si $f_1(x, z) = \min_{y \in D_Y} f(x, z, y)$ alors $\forall z \in D_{\mathbf{Z}}$ $\exists k \in D_Y$ tel que $\forall x \in D_X$, $f_1(x, z) = f(x, z, k)$.

Démonstration. Soit $z \in D_{\mathbf{Z}}$.

$\boxed{\exists x \in D_x,\ \exists y,\ f(x, z, y) \neq \top}$
Soit $D_K = \{k_1, k_2, \ldots, k_{d_Y}\}$ tel que $\forall i < j$, $f(x, z, k_i) \leqslant f(x, z, k_j)$. Nous en déduisons $\forall i \in [1, d_Y]$, $f(x, z, k_1) \leqslant f(x, z, k_i)$ et $f_1(x, z) = \min_{y \in D_Y} f(x, z, y) = f(x, z, k_1)$.
En utilisant le théorème 2, nous trouvons $\forall k_p \in D_K$ $\forall u \in D_X$, $0 \leqslant f(x, z, k_p) \boxminus f(x, z, k_1) \stackrel{\circ}{=} f(u, z, k_p) \boxminus f(u, z, k_1)$. Donc $f(u, z, k_1) \leqslant f(u, z, k_p)$ ou $f(u, z, k_1) = f(u, z, k_p) = \top$, ainsi $\forall u \in D_X, f_1(x, z) = \min_{y \in D_Y} f(u, z, y) = f(u, z, k_1)$.

$\boxed{\forall x \in D_x,\ \forall y,\ f(x, z, y) = \top}$
$\forall x \in D_X$, $f_1(x, z) = \min_{y \in D_Y} f(x, z, y) = \top$, donc soit $k \in D_y$, $\forall x \in D_X$, $f_1(x, z) = f(x, z, k) = \top$ □

Maintenant nous pouvons prouver le théorème 3.
Nous avons $f_1(x, z) = \min_{y \in D_Y} f(x, z, y)$ et $f_2(z, y) = \min_{x \in D_X} [f(x, z, y) - f_1(x, z)]$ donc $f_2(z, y) \leqslant f(x, z, y) - f_1(x, z)$.

Si $f_2(z, y) = f(x, z, y) - f_1(x, z)$, $f(x, z, y) = f_1(x, z) + f_2(z, y)$.

Si $f_2(z, y) < f(x, z, y) - f_1(x, z)$,

$\boxed{f(x, z, y) = f_1(x, z) = \top}$
Nous avons $f_1(x, z) + \min_{x \in D_X} [f(x, z, y) - f_1(x, z)] = \top = f(x, z, y)$ donc$f(x, z, y) = f_1(x, z) + f_2(z, y)$.

$\boxed{f(x, z, y) \neq \top \textbf{ ou } f_1(x, z) \neq \top}$
Nous avons $\min_{u \in D_X} [f(u, z, y) - f_1(u, z)] < f(x, z, y) - f_1(x, z)$ donc $f(u, z, y) - f_1(u, z) < f(x, z, y) - f_1(x, z)$ avec $u = \text{argmin}_{u \in D_X} [f(u, z, y) - f_1(u, z)]$. En utilisant le lemme 2, nous avons $l = \text{argmin}_{k \in D_Y} f(x, z, k) = \text{argmin}_{k \in D_Y} f(u, z, k)$,

ainsi $f(u,z,l) - f(u,z,l) < f(x,z,y) - f(x,z,l)$, mais également $f(u,z,y) \boxminus f(u,z,l) \doteq f(x,z,y) \boxminus f(x,z,l)$ car f est décomposable par paire par rapport à X, Y. De plus, $f(x,z,y) \neq \top$ ou $f(x,z,l) \neq \top$, ainsi $f(u,z,y) - f(u,z,l) = f(x,z,y) - f(x,z,l)$. Finalement $f_2(z,y) = f(x,z,y) - f_1(x,z)$, ainsi $f(x,z,y) = f_1(x,z) + f_2(z,y)$.

Références

[1] M. Cooper, S. de Givry, M. Sanchez, T. Schiex, M. Zytnicki, and T. Werner. Soft arc consistency revisited. *Artificial Intelligence*, 174(7–8) :449–478, 2010.

[2] S. de Givry, J. Larrosa, P. Meseguer, and T. Schiex. Solving max-sat as weighted csp. In *Proc. of CP-03*, pages 363–376, Kinsale, Ireland, 2003.

[3] M Fishelson, N Dovgolevsky, and D Geiger. Maximum likelihood haplotyping for general pedigrees. *Human Heredity*, 59 :41–60, 2005.

[4] J Hammersley and P Clifford. Markov fields on finite graphs and lattices, 1971.

[5] D. Heckerman and J. Breese. Causal independence for probability assessment and inference using bayesian networks. *IEEE Systems, Man, and Cyber.*, 26(6) :826–831, 1996.

[6] D. Koller and N. Friedman. *Probabilistic Graphical Models*. The MIT Press, 2009.

[7] J. Larrosa. Boosting search with variable elimination. In *CP*, pages 291–305, Singapore, 2000.

[8] S. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

[9] C. Lecoutre, L Saïs, S. Tabary, and V. Vidal. Reasoning from last conflict(s) in constraint programming. *Artificial Intelligence*, 173 :1592,1614, 2009.

[10] R. Marinescu and R. Dechter. Memory intensive and/or search for combinatorial optimization in graphical models. *Artificial Intelligence*, 173(16-17) :1492–1524, 2009.

[11] I. Meiri, R. Dechter, and J. Pearl. Tree decomposition with applications to constraint processing. In *Proc. of AAAI'90*, pages 10–16, Boston, MA, 1990.

[12] P. Meseguer, F. Rossi, and T. Schiex. Soft constraints processing. In *Handbook of Constraint Programming*, chapter 9. Elsevier, 2006.

[13] M. Sánchez, S. de Givry, and T. Schiex. Mendelian error detection in complex pedigrees using weighted constraint satisfaction techniques. *Constraints*, 13(1) :130–154, 2008.

[14] M. Sánchez, P. Meseguer, and J. Larrosa. Using constraints with memory to implement variable elimination. In *ECAI*, pages 216–220, Spain, 2004.

[15] P. Savicky and J. Vomlel. Exploiting tensor rank-one decomposition in probabilistic inference. *Kybernetika*, 43(5) :747–764, 2007.