2-triangulation de micro-structure pour la résolution de CSP

Samba Ndojh Ndiaye

2-triangulation de micro-structure pour la résolution de CSP

Samba Ndojh NDIAYE
LSIS - UMR CNRS 6168
Université Paul Cézanne (Aix-Marseille III)
Avenue Escadrille Normandie-Niemen
13397 Marseille Cedex 20 (France)
samba.ndiaye@lsis.org

Résumé : Un CSP ou problème de satisfaction de contraintes, consiste à donner des valeurs à des variables en respectant les contraintes qui les lient. Mais le problème de décision associé à un CSP est NP-complet.
Dans [Jégou, 1993], Jégou propose une méthode, basée sur la décomposition de la micro-structure du CSP et dans [Chmeiss, 2003], une généralisation de cette méthode est présentée en utilisant une généralisation des graphes triangulés : les graphes CSG^k.
Nous proposons une amélioration de la décomposition de CSP basée sur les graphes CSG^2. Cela passe par un calcul de 2-triangulation plus efficace tant au niveau de la complexité qu'à celui de la qualité de la décomposition, mais aussi un algorithme de recherche de cliques maximales dans les graphes 2-triangulés, apportant de bons résultats en pratique.
Mots-clés : Intelligence artificielle, satisfaction de contraintes, algorithmique des graphes.

1 INTRODUCTION

Un CSP ou problème de satisfaction de contraintes, consiste à donner des valeurs à des variables en respectant les contraintes qui les lient. Il existe un grand nombre d'applications à cela, notamment en intelligence artificielle avec la configuration, la planification, la coloration de graphe, par exemple et des problèmes de combinatoire comme celui des n-dames, etc. Mais le problème de décision associé à un CSP c'est-à-dire de l'existence d'une solution est NP-complet et la méthode de résolution standard, le backtrack, provoque une explosion combinatoire. C'est la raison pour laquelle de nombreuses études ont été faites pour l'améliorer et réduire ainsi sa complexité. Dans [Jégou, 1993], une méthode nouvelle, différente des approches prônées jusque-là, nous est proposée. Elle est basée sur la décomposition de la micro-structure du CSP qui est un graphe dont les sommets sont des couples (variable, valeur) et les arêtes sont définies par les relations de compatibilité induites par les contraintes entre ces couples. En recherchant les cliques maximales dans le graphe triangulé de la micro-structure, on divise le problème en plusieurs sous-problèmes indépendants avec une complexité de résolution au plus égale à celle du problème initial. Dans [Chmeiss, 2003], une généralisation de cette méthode est présentée en utilisant une généralisation des graphes triangulés : les graphes CSG^k qui ont quelques propriétés intéressantes en commun avec les graphes triangulés, notamment des algorithmes polynomiaux de résolution de problèmes NP-difficiles tel que la recherche des cliques maximales. Elle permet de mieux décomposer le problème et de contrôler cette décomposition avec le paramètre k. Mais cette décomposition se heurte aux coûts très élevés des algorithmes de 2-triangulation et de recherche des cliques maximales. Une amélioration de cette décomposition passe donc par la mise en oeuvre d'algorithmes de manipulations des graphes CSG^k plus performants. Dans cette optique, nous proposons dans le cadre des graphes CSG^2, un algorithme de 2-triangulation, plus efficace tant au niveau de sa complexité qu'à celui du nombre d'arêtes rajoutées, de même qu'un algorithme de recherche des cliques maximales dans ces graphes qui donne de très bons résultats.
Cet article va commencer par un rappel du formalisme CSP et de quelques méthodes de résolutions les plus connues. Dans la deuxième partie, nous parlerons des méthodes de décomposition de CSP proposée dans [Jégou, 1993] et [Chmeiss, 2003], avant de présenter les améliorations que nous proposons et les résultats expérimentaux obtenus dans les sections 3 et 4.

2 RAPPELS

2.1 Problème de satisfaction de containtes : CSP

Un CSP est la donnée d'un quadruplet (X, D, C, R) :
- $X = \{X_1, \ldots, X_n\}$ est un ensemble de variables ;
- $D = \{D_1, \ldots, D_n\}$ est un ensemble de domaines, chaque domaine D_i étant associé à la variable X_i ;
- $C = \{C_1, \ldots, C_m\}$ est un ensemble de contraintes, chaque contrainte C_i portant sur un sous-ensemble de X, $\{X_{i_1}, \ldots, X_{i_{n_i}}\}$;
- $R = \{R_1, \ldots, R_m\}$ est un ensemble de relations, chaque relation R_i étant des combinaisons de valeurs satisfaisant C_i donc R_i est un sous-ensemble de $D_{i_1} \times \ldots \times D_{i_{n_i}}$.

Un CSP est dit binaire si les contraintes portent au plus sur deux variables.
Un graphe est le couple (V, E) avec V, les sommets du graphe, E, ses arêtes qui relient deux sommets.

Définition 2.1.1 : Le graphe de contraintes d'un CSP binaire (X, D, C, R) est le graphe (X, C), X étant l'ensemble des sommets et C l'ensemble des arêtes.

Une solution d'un CSP est une affectation de toutes les variables qui satisfait toutes les contraintes. Le problème de décision est NP-complet, même si on se restreint aux CSP binaires. La méthode standard de résolution d'un CSP est le backtrack. A chaque étape, on cherche à étendre l'instanciation partielle en affectant une valeur consistante avec l'instanciation partielle à une nouvelle variable. Cette approche est fortement combinatoire avec une taille de l'espace de recherche en $O(d^n)$ où d est la taille maximale des domaines et n, le nombre de variables. Face à l'inefficacité du backtrack, des algorithmes très performants ont été développés tels que le Forward Checking [Haralick, 1980], MAC [Sabin, 1994]. Nous nous intéressons ici à une méthode basée sur la triangulation et la décomposition de la micro-structure des CSP et une généralisation de cette méthode : la k-triangulation avec les graphes CSG^k .

Dans [Jégou, 1993], Jégou nous donne une méthode alternative de résolution de CSP qui se base sur la décomposition de la micro-structure.
Dans toute la suite on va se restreindre aux CSP binaires. En outre, on va noter une contrainte portant sur les variables v_i et v_j par C_{ij}. Nous allons également utiliser la notation $G_{\{v_i,\ldots,v_n\}}$ pour $G = (V, E)$ un graphe, qui désigne le graphe restreint aux sommets $v_i, \ldots, v_n$ de V, avec pour ensemble de sommets : $V' = \{v_i, \ldots, v_n\}$ et pour ensemble d'arêtes : $E' = \{C_{ij} \in E / v_i \in V'$ et $v_j \in V'\}$.

Définition 2.1.2 : Soit $\mathcal{P} = (X, D, C, R)$ un CSP tel que son graphe de contraintes (X, C) soit complet, on appelle $\mu(\mathcal{P})$ la micro-structure de $\mathcal{P}$ le graphe n-parties défini comme suit :

- $X_D = \{(X_i, a) / X_i \in X, a \in D_i\}$;
- $C_R = \{\{(X_i, a), (X_j, b)\} / (X_i, X_j) = C_{ij} \in C, (a, b) \in R_{ij}\}$;
- $\mu(\mathcal{P}) = (X_D, C_R)$.

La micro-structure est donc le graphe avec comme sommets des couples (variable, valeur), valeur étant dans le domaine de la variable, et il existe une arête entre deux sommets si leurs valeurs respectives sont autorisées par la contrainte qui les lient.

Exemple : $\mathcal{P} = (X, D, C, R)$

- $X = \{X_1, X_2, X_3, X_4\}$;
- $D = \{D_1, D_2, D_3, D_4\}$ $D_1 = \{a, b\}$ $D_2 = \{c, d\}$ $D_3 = \{e, f\}$ $D_4 = \{g, h\}$;
- $C = \{C_{12}, C_{13}, C_{14}, C_{23}, C_{24}, C_{34}\}$;
- $R = \{R_{12}, R_{13}, R_{14}, R_{23}, R_{24}, R_{34}\}$.

R12		*R13*		*R14*		*R23*		*R24*		*R34*	
X1	X2	X1	X3	X1	X4	X2	X3	X2	X4	X3	X4
a	c	b	e	a	h	c	e	c	g	e	g
b	c			b	g	d	e			e	h
b	d					d	f			f	g

Tableau 1 : *Les relations définies par les contraintes.*

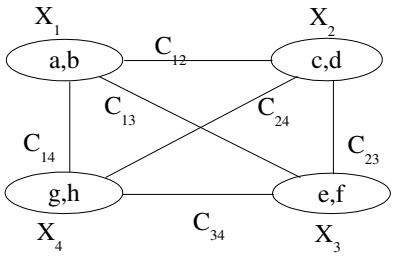

Figure 1 : *Graphe de contraintes complet.*

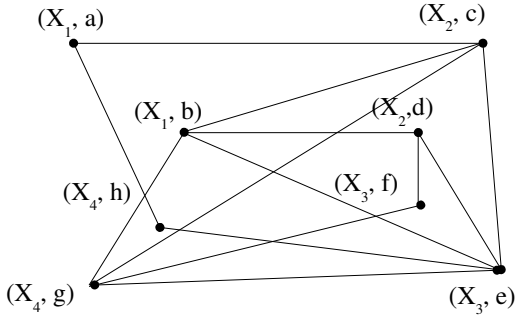

Figure 2 : *Micro-structure de $\mathcal{P}$.*

Ce graphe est nécessairement n-parties car il ne peut pas exister d'arête entre deux sommets issus du domaine d'une même variable. Si (X, C) n'est pas complet, il faut rajouter la contrainte universelle entre les variables qui n'ont pas de contraintes entre elles. La relation associée à la contrainte universelle entre deux variables X_i et X_j est $R_{ij} = D_i \times D_j$ c'est-à-dire que tous les couples de valeurs sont compatibles. On peut donc se contenter de ne considérer que les graphes de contraintes complets.

Proposition 2.1.3 [Jégou, 1993] : Soit $\mathcal{P} = (X, D, C, R)$ un CSP et $\mu(\mathcal{P})$ sa micro-structure : $(a_1, \ldots, a_n)$ est solution de $\mathcal{P}$ $\iff$ $\{(X_1, a_1), \ldots, (X_n, a_n)\}$ est une n-clique (clique de taille n) de $\mu(\mathcal{P})$.

Une solution de $\mathcal{P}$ constitue donc un recouvrement des n sommets (variables) du graphe des contraintes (X, C). La résolution d'un CSP se ramène ainsi à la recherche des n-cliques dans sa micro-structure. Mais le problème de recherche des n-cliques dans un graphe est NP-complet. L'idée est donc d'appliquer la méthode sur des micro-structures particulières pour lesquelles il existe des algorithmes polynomiaux de recherche des cliques maximales. Les graphes triangulés en sont un exemple.

2.2 Graphes triangulés et CSG^k

Dans [Golumbic, 1980], on trouve une présentation assez complète des graphes triangulés et de leurs propriétés. Nous rappelons cependant ici quelques notions essentielles.

Définition 2.2.1 : Un graphe G est triangulé si tout cycle de longueur supérieure strictement à 3 possède une corde c'est-à-dire une arête qui joint deux sommets non consécutifs.

On peut aussi caractériser les graphes triangulés par la notion de sommet simplicial.

Définition 2.2.2 : Un sommet v d'un graphe G est simplicial si l'ensemble de ses voisins induit un graphe complet.

Un ordre $\sigma = [v_1, \ldots, v_n]$ est un schéma parfait d'élimination si chaque v_i est simplicial pour le sous-graphe induit par $G_{\{v_i,\ldots,v_n\}}$ c'est-à-dire $\{v_j \in Adj(v_i)/j > i\}$ est complet, $Adj(v_i)$ étant l'ensemble des sommets voisins de v_i.

Théorème 2.2.3 [Fulkerson, 1965] : Soit G un graphe, G est triangulé ssi G possède un schéma parfait d'élimination.

Les graphes triangulés ont quelques propriétés très utiles. En effet, il existe des algorithmes polynomiaux pour la résolution de problèmes NP-complets tels que : clique, stable, coloriage, recouvrement par un ensemble de cliques.

Proposition 2.2.4 [Fulkerson, 1965] : Un graphe triangulé avec n sommets a au plus n cliques maximales.

Cette proposition permet l'élaboration d'un algorithme qui étant donné un graphe triangulé, calcule ses cliques maximales en un temps linéaire.

Proposition 2.2.5 [Gavril, 1972] : La recherche des cliques maximales dans un graphe triangulé $G = (V, E)$ est en $O(n + m)$ avec $n = |V|$ et $m = |E|$.

Dans la section précedente, on a vu que la résolution d'un CSP revenait à la recherche de n-cliques dans sa micro-structure, un problème NP-difficile. Or il existe un algorithme linéaire pour les graphes triangulés. D'où l'idée dans [Jégou, 1993] de procéder à la triangulation de cette micro-structure, c'est-à-dire lui rajouter des arêtes pour la transformer en graphe triangulé. Il existe des algorithmes de triangulation linéaires [Kjaerulff, 1990]. Néanmoins, avoir une triangulation optimale (qui minimise la taille de la plus grande clique et donc des sous-problèmes) est un problème NP-difficile [Arnborg, 1987]. D'où la nécessité de se restreindre à une approximation de cette optimalité. Vu les propriétés intéressantes des graphes triangulés dans la méthode de résolution des CSP basée sur la micro-structure, on voit rapidement l'apport que pourrait représenter une généralisation de ces graphes avec les mêmes propriétés. Cela permettrait de travailler sur un ensemble de graphes de taille plus considérable, mais aussi d'éviter la trop grande restriction que constitue la triangulation. Dans [Chmeiss, 1997], une généralisation est donnée, les graphes CSG^k, qui sont définis de manière inductive.

Définition 2.2.6 : Les graphes CSG^0 forment la classe des graphes complets. Pour $k > 0$, les graphes CSG^k sont la classe des graphes $G = (V, E)$ tel qu'il existe un ordre $\sigma = [v_1, \ldots, v_n]$ sur V tel que pour $i = 1, \ldots, n$ le graphe $G(N^+(v_i))$ est CSG^{k-1} où $N^+(v_i) = \{v_j \in V/\{v_i, v_j\} \in E, i < j\}$.
$G(N^+(v_i))$ est le graphe restreint aux sommets de $N^+(v_i)$. L'ordre σ est appelé schéma CSG^k .

Proposition 2.2.7 [Chmeiss, 1997] : Les graphes CSG^1 forment la classe des graphes triangulés.
Proposition 2.2.8 [Chmeiss, 1997] : Pour tout $k \geq 0$, tout graphe CSG^k est aussi CSG^{k+1}.
Théorème 2.2.9 [Chmeiss, 1997] : Tout sous-graphe d'un graphe CSG^k est CSG^k .
Théorème 2.2.10 [Chmeiss, 1997] : Un graphe CSG^k avec n sommets a au plus n^k cliques maximales.

Les graphes CSG^k sont donc bien une généralisation des graphes triangulés.
Dans [Chmeiss, 1997], il est donné un algorithme de recherche de la clique de taille maximale dans les graphes CSG^k avec une complexité en $O(n^{2(k-1)}(n + m))$.
Pour le cas des graphes CSG^2 nous avons un résultat supplémentaire.

Théorème 2.2.11 [Chmeiss, 1997] : Un graphe CSG^2 avec n sommets a au plus $(n + m)$ cliques maximales.

Tous ces résultats sur les graphes CSG^k montrent l'intérêt que présente cette généralisation dans la mesure où on retrouve plusieurs bonnes propriétés notamment un nombre de cliques maximales limité et un algorithme polynomial de recherche de ces cliques.

2.3 La TRk-Décomposition

Etant donnée la micro-structure d'un CSP, *a priori* elle n'est pas CSG^k d'où pour pouvoir utiliser un algorithme polynomial de recherche de cliques maximales pour les graphes CSG^k, la nécessité de la rendre CSG^k. On appelle k-triangulation le processus qui consiste à rajouter des arêtes pour rendre un graphe quelconque CSG^k. On note $\Gamma(X_D, C_R)$ le graphe k-triangulé de $\mu(\mathcal{P})$. $\Gamma(X_D, C_R)$ a au plus $(n.d)^k$ cliques maximales et il existe un algorithme polynomial pour les calculer. On sait que si des solutions existent, elles seront dans les cliques maximales donc leur recherche peut se limiter à ces cliques. On a une décomposition du problème en plusieurs sous-problèmes. Soit Y une clique maximale de $\Gamma(X_D, C_R)$:

- Si Y ne recouvre pas toutes les variables de $\mathcal{P}$ alors elle ne contient pas de solution ;
- Si Y recouvre toutes les variables, on n'est pas sûr de tenir une solution car la k-triangulation a rajouté des arêtes entre des sommets qui sont incompatibles. On crée donc un nouveau CSP en projetant les sommets de Y dans chaque domaine de chaque variable. On obtient de nouveaux domaines $D_{Y,i} \subset D_i$ avec $a \in D_{Y,i}$ si et seulement si $(X_i, a) \in Y$. Et on

garde les contraintes de $\mathcal{P}$ restreintes aux nouveaux domaines des variables. La recherche de solution se fait sur le nouveau CSP.

Définition 2.3.1 : Soient $\mathcal{P} = (X, D, C, R)$ un CSP, $\mu(\mathcal{P}) = (X_D, C_R)$ sa micro-structure et Y un sous-ensemble de X_D. Le CSP induit par Y dans $\mathcal{P}$, noté $\mathcal{P}(Y)$, est défini comme suit :
- $D_Y = \{D_{Y,1}, \ldots, D_{Y,n}\}$ tel que $D_{Y,i} = \{a \in D_i / (X_i, a) \in Y\}$
- $R_{Y,ij} = \{(a,b) \in R_{ij} / \{(X_i, a), (X_j, b)\} \in Y\}$
- $\mathcal{P}(Y) = (X, D_Y, C, R_Y)$

Exemple : Nous reprenons notre exemple et procédons à la triangulation $(k = 1)$ de la micro-structure.

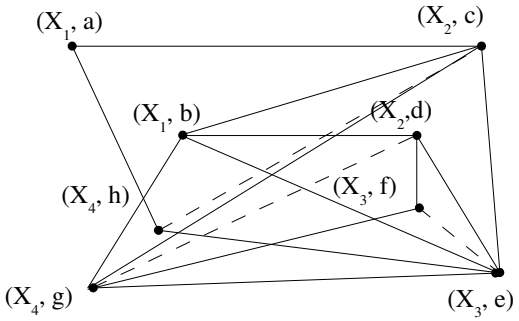

Figure 3 : *Triangulation de la micro-structure de $\mathcal{P}$.*

Cliques maximales :
$Y_1 = \{(X_1, a), (X_2, c), (X_4, h)\}$
$Y_2 = \{(X_1, b), (X_2, c), (X_3, e), (X_4, g)\}$
$Y_3 = \{(X_1, b), (X_2, d), (X_3, e), (X_4, g)\}$
$Y_4 = \{(X_2, c), (X_3, e), (X_4, h)\}$
$Y_5 = \{(X_2, d), (X_3, e), (X_3, f), (X_4, g)\}$

Décomposition des domaines :
$D_{Y_{1,1}} = \{a\}, D_{Y_{1,2}} = \{c\}, D_{Y_{1,3}} = \emptyset, D_{Y_{1,4}} = \{h\}$
$D_{Y_{2,1}} = \{b\}, D_{Y_{2,2}} = \{c\}, D_{Y_{2,3}} = \{e\}, D_{Y_{2,4}} = \{g\}$
$D_{Y_{3,1}} = \{b\}, D_{Y_{3,2}} = \{d\}, D_{Y_{3,3}} = \{e\}, D_{Y_{3,4}} = \{g\}$
$D_{Y_{4,1}} = \emptyset, D_{Y_{4,2}} = \{c\}, D_{Y_{4,3}} = \{e\}, D_{Y_{4,4}} = \{h\}$
$D_{Y_{5,1}} = \emptyset, D_{Y_{7,2}} = \{d\}, D_{Y_{7,3}} = \{e, f\}, D_{Y_{7,4}} = \{g\}$

Pour chaque clique maximale, on définit un nouveau CSP avec de nouveaux domaines issus de leur décomposition. On voit que les cliques Y_1, Y_4, Y_5 ne recouvrent pas toutes les variables et les CSP induits ont au moins un domaine vide, de ce fait ne peuvent pas contenir une solution du CSP. Par contre Y_2 et Y_3 recouvrent toutes les variables et le CSP induit a des domaines de taille 1 et est consistant pour Y_2, et inconsistant pour Y_3.

Théorème 2.3.2 : Soient $\mathcal{P}$ un CSP, $\mu(\mathcal{P})$ sa micro-structure, $\Gamma(\mu(\mathcal{P}))$ la k-triangulation de cette dernière et $Y = \{Y_1, \ldots, Y_p\}$ l'ensemble des cliques maximales de $\Gamma(\mu(\mathcal{P}))$ alors :
$Solutions(\mathcal{P}) = \bigcup_{1 \leq i \leq p} Solutions(\mathcal{P}(Y_i))$.

Comme cela a été fait pour les graphes triangulés, la même démarche est utilisée dans [Chmeiss, 2003]. Cette démarche reposant sur les cliques maximales de la micro-structure du CSP, il est utile d'en avoir un nombre qui ne soit pas exponentiel pour que la résolution soit réalisable en pratique. Ce qui est le cas des graphes CSG^k. On reprend l'algorithme de résolution de CSP par décomposition des domaines

TR^k-Décomposition :

1. Construction de $\mu(\mathcal{P})$
2. k-Triangulation de $\mu(\mathcal{P})$: $\Gamma(\mu(\mathcal{P}))$
3. Recherche de toutes les cliques maximales de $\Gamma(\mu(\mathcal{P}))$: $Y = \{Y_1, \ldots, Y_p\}$
4. Pour tout $Y_i \in Y$ faire
   Si Y_i recouvre toutes les variables
   Alors Résoudre $\mathcal{P}(Y_i)$
   Sinon $\mathcal{P}(Y_i)$ n'a pas de solution.

Soit p, le nombre de cliques maximales dans $\Gamma(\mu(\mathcal{P}))$ et soit δ tel que $|D_{Y_{j,i}}| \leq \delta$ pour $i = 1, \ldots, n$ $j = 1, \ldots, p$. La complexité de la TR^1-Décomposition est en $O(m.p.\delta^n)$ contre $O(md^n)$ pour le backtrack standard.
Dans [Chmeiss, 2003], il est suggéré un remplacement de l'itération de la ligne 4 par une exécution en parallèle de la résolution des différents sous-problèmes étant donné qu'ils sont tous indépendants. Dans [Habbas, 2000], une telle implémentation est présentée pour la TR-décomposition et le coût effectif de résolution d'un CSP consistant sera le coût de résolution du problème le plus simple et pour un CSP inconsistant celui du problème le plus difficile. Donc cette même résolution en parallèle des sous-problèmes peut être réalisée dans le cas de la TR^2-décomposition avec les graphes 2-triangulés.
Pour avoir une bonne décomposition des domaines (avec un petit nombre de valeurs dans chaque domaine des sous-problèmes), il est nécessaire d'avoir un bon algorithme de k-triangulation. Chmeiss, Jégou et Keddar ont élaboré un algorithme de 2-triangulation et le fait de se restreindre au cas $k = 2$ est dû au coût relativement important de manipulation des graphes CSG^k (reconnaissance, recherche des cliques en $O(n^{2(k-1)}(n+m))$. Cet algorithme, avec une complexité théorique en $O(nm(n+m))$, n'a pas permis de traiter des jeux de données de taille satisfaisante.

Toutefois, dans [Chmeiss, 2003], une étude comparative a été menée pour montrer l'utilité pratique de la méthode de décomposition généralisée et par triangulation par rapport à MAC qui est avec FC l'un des meilleurs algorithmes de résolution de CSP. Pour la décomposition généralisée, seul le cas $k = 2$ est étudié. Les résultats obtenus montrent que TR^1-Décomposition et TR^2-Décomposition améliorent de manière très significative les performances de MAC sur toutes les classes de CSP traitées et surtout pour des problèmes à taille de domaines très grande. On observe aussi que TR^2 est beaucoup plus

efficace que TR[1] avec un nombre de sous-problèmes plus important et une meilleure décomposition du problème, ce qui conduit à une résolution plus rapide. Mais ces résultats sont limités à cause du coût de la k-triangulation et de la recherche de ses cliques maximales, qui sont rapidement prohibitifs. Pour rendre effectivement opérationnelle la décomposition, il était nécessaire d'élaborer des algorithmes de 2-triangulation et de recherche de cliques maximales beaucoup plus efficaces pour réduire leur coût tout en améliorant la qualité de la décomposition.

3 ALGORITHMES DE LA TR2-DÉCOMPOSITION

Ici, nous allons présenter essentiellement un algorithme de 2-triangulation et un de recherche de cliques maximales dans les graphes ainsi 2-triangulés qui permettent d'améliorer la TR^2-décomposition. La 2-triangulation d'un graphe G consiste principalement à choisir un ordre sur les sommets et à rajouter des arêtes tel que pour tout v_i sommet de G, $G(N^+(v_i))$ soit triangulé. On cherche à s'approcher de la 2-triangulation optimale qui induit une taille de clique maximale la plus petite possible et qui donnerait des sous-problèmes avec des tailles de domaines plus petites et donc plus faciles à résoudre. Mais, l'efficacité est primordiale et nous impose donc de faire un compromis avec l'optimalité, d'où l'algorithme de 2-triangulation en $O(nm(n+m))$ utilisé dans [Chmeiss, 2003] pour la TR^2-Décomposition. L'objectif est donc d'élaborer un algorithme qui va trouver un bon compromis entre l'approximation de l'optimalité et une complexité raisonnable. Une procédure de recherche de cliques maximales plus efficace dans la TR^2-Décomposition devrait également améliorer de manière significative cette dernière. Le but de ce travail est autant d'avoir une meilleure complexité théorique que d'avoir des performances chronométriques qui prouvent l'intérêt pratique de cette approche.

3.1 Un algorithme de 2-triangulation

La principale difficulté dans l'élaboration d'un algorithme de 2-triangulation est la validité : est-ce que le graphe obtenu est 2-triangulé ?
La définition même de la 2-triangulation nous donne une première approche qui va consister à prendre un ordre quelconque sur les sommets du graphe σ, et pour $i = 1$ à n, trianguler $G(N^+((v_{\sigma(i)})))$, et cette triangulation n'impose pas d'ordre sur les sommets de $G(N^+((v_{\sigma(i)})))$. On peut donc choisir un ordre arbitraire σ', pour l'effectuer. Mais la triangulation d'un sous-graphe peut remettre en cause la triangulation d'un sous-graphe antérieur, ce qui donne en sortie un graphe qui n'est pas 2-triangulé.
Ce problème ne se pose pas pour la triangulation car la complétion d'un sous-graphe induit ne remet pas en cause la complétude des sous-graphes antérieurs. En imposant le même ordre pour la 2-triangulation et la triangulation des sous-graphes induits, il est plus facile de circonscrire la difficulté et ainsi définir un algorithme efficace.
L'ordre de la 2-triangulation peut-être quelconque, une fois choisie on se propose de la transformer en un ordre $CSG^{2,1}$ qui est défini comme suit :

Définition 3.1.1 : Un ordre est dit $CSG^{2,1}$ s'il est CSG^2 et pour tout $i = 1$ à n $G(N^+((v_{\sigma(i)})))$ est triangulé suivant l'ordre σ.

La deuxième condition n'est pas nécessaire dans la 2-triangulation, mais en imposant cette restriction, il est plus facile d'appréhender les cas de remises en cause d'une triangulation d'un sous-graphe induit à une étape ultérieure.

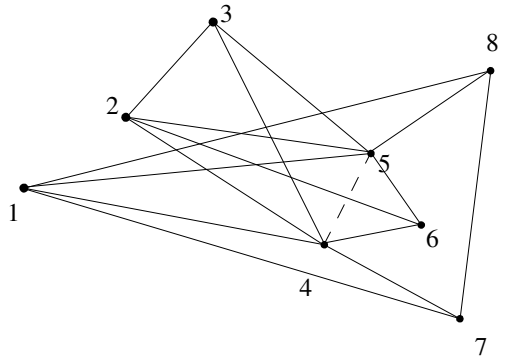

Figure 4 : *Cas de remise en cause.*

La triangulation de $G(N^+(2))$ rajoute une arête entre 4 et 5 et crée ainsi un cycle de longueur 4 dans $G(N^+(1))$. Il y a donc remise en cause de la triangulation de $G(N^+(1))$.

Définition 3.1.2 : On appelle cas gênant CG, un graphe à 4 sommets formant un cycle de longueur 4 privé d'une arête.

Le quadruplet $\{2, 3, 4, 5\}$ constitue un cas gênant qui mène à une remise en cause d'une triangulation antérieure.
Théorème 3.1.3 : La triangulation d'un sous-graphe $G(N^+((v_j)))$ rajoute une arête à un sous graphe $G(N^+((v_i)))$, v_i précédant v_j dans l'ordre si et seulement si on a la configuration de la figure 5 à un renommage des sommets près.
preuve : Soient G un graphe et v_2 un sommet tel que la triangulation de $G(N^+((v_2)))$ rajoute une arête à $G(N^+((v_1)))$, v_1 précède v_2 dans l'ordre. On note v_4 et v_5 les deux sommets entre lesquels cette arête est rajoutée. Et pour qu'elle soit rajoutée, il faut l'existence dans $G(N^+((v_2)))$ d'un sommet v_3 adjacent à v_4 et v_5 et qui les précéde dans l'ordre. On retrouve exactement la configuration de la figure 5. CQFD

L'idée de l'algorithme est basée sur ce théorème, elle consiste à repérer tous les quadruplets de sommets v_2, v_3 , v_4 , v_5 et de rajouter l'arête entre v_4 et v_5 de façon préventive, afin d'éviter toute remise en cause.

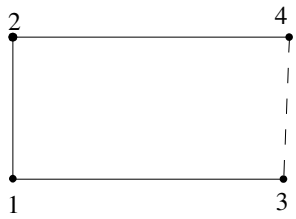

Figure 5 : *Cas gênant.*

Nous proposons un algorithme appelé Deuxtriang qui élimine tous les cas gênants en parcourant le graphe une seule fois. Cet algorithme utilise un ordre quelconque fourni en entrée.

Deuxtriang (G)
Entrées : graphe G, ordre σ
Sorties : graphe Ge=(V, E')

Ge $<-$ G
pour i=3 a n-1
  pour j=i+1 a n si (v_i non adjacent v_j)
    pour k=1 a i-1 si (v_k adjacent v_i et v_k adjacent v_j)
      pour l=1 a i-1 si (k !=l et v_l adjacent v_i et v_l adjacent v_j)
        si (v_l adjacent v_k)
        rendre v_i et v_j adjacent

L'algorithme repère les cas gênants en partant des sommets non adjacents et pour chaque paire de sommets non adjacents, on regarde s'ils ont deux prédécesseurs adjacents en commun, dans quels cas il faut rajouter l'arête manquante.

Théorème 3.1.4 : ***Deuxtriang*** supprime tous les cas gênants.
preuve : On note ***Ge*** le graphe à la sortie de l'algorithme. Evidemment ce dernier s'arrête, mais il faut vérifier qu'il n'y a pas de cas gênants dans ***Ge***.
On va faire une récurrence sur i. Pour $i = 3$, il n'y a pas de cas gênants. Considérons un i donné. On suppose que les cas gênants antérieurs à v_i ont déjà été traités. Considérons l'ajout d'arêtes de la forme v_i,v_j. Il s'agit d'arêtes qui ne peuvent rajouter de cas gênant antérieur à v_i. En effet, l'ajout d'une arête crée un nouveau cas si une des arêtes manquait et qu'elle est rajoutée et étant donné que les deux sommets non adjacents suivent les autres dans l'ordre, donc ce cas n'a pas encore été examiné et le sera forcément.
A l'étape $i = n$, tous les cas ont été résolus. CQFD

Théorème 3.1.5 : ***deuxtriang*** calcule la 2-triangulation du graphe G, donné.
preuve : On prend dans ***Ge*** un sommet v_{i_1} tel que le sous-graphe induit possède deux sommets non adjacents v_{i_3} et v_{i_4}, si la triangulation de $G(N^+(v_{i_1}))$ nécessite de rajouter une arête entre v_{i_3} et v_{i_4}, il existe donc un sommet v_{i_2} qui les précéde et qui est adjacent à la fois à v_{i_1} et à v_{i_3} et v_{i_4}. Cette description correspond à un cas gênant, or il n'en existe pas dans ***Ge***. Il n'est pas nécessaire donc de rajouter des arêtes pour qu'il soit 2-triangulé. CQFD

Corollaire 3.1.6 : ***Deuxtriang*** est correcte.
Ce corollaire découle directement du théorème précédent.

Théorème 3.1.7 : La complexité de ***Deuxtriang*** est en $O(n^4)$.

N=1000		***Deuxtriang_MCS***			***Deuxtriang_AB***			***triang_ord***			***triang_min***		
m		durée	clic	ma	durée	clic	ma	durée	clic	ma	durée	clic	ma
10,00%	max		854			837			938	431641		912	
	moy	13s97	215	372582	6s06	151	354204	0s21	669		9s34	571	415756
	nbcm		1143			1159			59			87	
30,00%	max		964			957			979			968	
	moy	13s65	473	338258	6s07	375	331707	0s18	790	345832	8s74	737	342019
	nbcm		197			204			20			32	
50,00%	max		981			978			989			982	
	moy	11s05	606	246172	5s19	540	243451	0s13	847	248088	7s30	815	246475
	nbcm		78			81			11			18	
70,00%	max		989			987			993			989	
	moy	7s92	755	147654	4s20	719	147654	0s09	889	149191	4s42	878	148636
	nbcm		34			36			7			11	
90,00%	max		995			993			997		2s02	994	
	moy	4s81	897	49712	3s13	889	49509	0s05	954	49800	1s32	948	49668
	nbcm		12			14			3			6	

Tableau 2 : n = 1000 sommets.

Deuxtriang_MCS : Deuxtriang avec ordre MCS + cliquesmax
Deuxtriang_AB : Deuxtriang avec ordre Anne Berry + cliquesmax
triang_ord : triang avec ordre MCS + cliquemax1
triang_min : triang minimale Anne Berry + cliquemax1
ma : nombre d'arêtes ajoutées
clic-max : taille clique maximale en moyenne
clic-moy : taille clique en moyenne

Sur un plan pratique cet algorithme a été comparé à deux algorithmes de triangulation : MCS [Tarjan, 1984] (***triang-ordre*** qui est linéaire) et AB, une triangulation minimale au sens de l'inclusion proposée par Anne Berry dans [Berry, 1999] (***triang-min***) sur des graphes générés aléatoirement. On a fait deux choix différents pour l'ordre de la 2-triangulation, l'ordre MCS et celui issu de AB, ce qui a conduit respectivement à la TR^2-Décomposition + MCS et TR^2-Décomposition + AB. On a pu constater que ***Deuxtriang*** a un temps d'exécution de même ordre de grandeur que la ***triang-min*** , et comme on pouvait l'espérer, il génére des cliques en moyenne beaucoup plus petites que les triangulations(jusqu'à deux fois plus petites).

3.2 Calcul des cliques maximales

Nous allons maintenant présenter un algorithme de recherche de cliques maximales dans les graphes 2-triangulés avec le même ordre pour la 2-triangulation et la triangulation des sous-graphes induits.
Notations : Etant donnés G un graphe CSG^2 et σ un ordre CSG$^{2.1}$ sur G, on note $C_{i,u}$ la clique avec i comme premier élément et u second, selon l'ordre σ, tel que $C_{i,u} - \{i\}$ est une clique maximale du sous-graphe $G(N^+[i])$.
Dans toute la suite, on va uniquement considérer les graphes 2-triangulés munis d'un ordre CSG$^{2.1}$.

Proposition 3.2.1 : Pour toute clique C de G, il existe i et u tels que $C \subseteq C_{i,u}$.
preuve : Soient C une clique de G et i son premier élément suivant l'ordre, $C - \{i\}$ est une clique de $G(N^+(i))$ qui est un sous-graphe triangulé. Il existe une clique maximale de $G(N^+(i))$ contenant $C - \{i\}$, u étant son premier élément suivant l'ordre, notée C_u^{max}.
On a : $C - \{i\} \subset C_{max,u}$ d'où $C \subset C_{max,u} \cup \{i\} = C_{i,u}$.

CQFD

Proposition 3.2.2 : Pour i et u fixés, $C_{i,u}$ est unique.
preuve : $G(N^+(i))$ étant triangulé, il existe une unique clique maximale dans $G(N^+(i))$ dont le premier élément est u, d'où $C_{i,u}$ est unique. CQFD

Proposition 3.2.3 : Si $C_{i,u}$ n'est pas maximale dans G alors il existe j tel que $C_{i,u} \subset C_{j,i}$.
preuve : On suppose $C_{i,u}$ non maximale alors il existe j et k tels que $C_{i,u} \subsetneq C_{j,k}$ donc $i \in C_{j,k}$ dont le premier élément dans l'ordre est j. On a alors $j \leq i$, mais si $i = j$ $C_{i,u} - \{i\} \subsetneq C_{i,k} - \{i\}$, or $C_{i,u} - \{i\}$ est une clique maximale dans $G(N^+(i))$. Donc $i < j$. On prend $j_0 = max\{j/C_{i,u} \subsetneq C_{j,k}\}$, $C_{i,u} \subsetneq C_{j_0,k_0} \Longrightarrow k_0 \leq i$. Si $k_0 < i$, on prend $k'_0 = max\{k'/k' \in C_{j_0,k_0} et k_0 \leq k' < i\}$ C_{j_0,k_0} étant une clique contenant $C_{i,u}$ et k'_0 alors $C_{i,u} \subset N^+(k'_0)$ d'où l'existence de $k"_0 \in N^+(k'_0)$ tel que $C_{i,u} \subsetneq C_{k'_0,k"_0}$ avec $j_0 < k_0 \leq k'_0 < i$ Contradiction avec la minimalité de j_0 donc $k_0 = i$. CQFD

On note :
$\mathcal{C}_m(G) = \{$ cliques maximales de $G\}$
$\mathcal{C}_m(G,i) = \{C_{i,u}/C_{i,u} - \{i\}$ cliques maximales de $G(N^+(i))\}$

Proposition 3.2.4 : $\mathcal{C}_m(G) \subset \bigcup_{1\leq i\leq n}(\mathcal{C}_m(G,i))$.
preuve : Soit $C \in \mathcal{C}_m(G)$, soit i le premier élément de C selon l'ordre et u, le second. $C - \{i\} \subset N^+(i)$: est une clique de $G(N^+(i))$ Si $C - \{i\}$ n'est pas maximale alors il existe C_{max} : clique maximale de $G(N^+(i))$ telle que $C - \{i\} \subset C_{max} \Longrightarrow C \subset C_{max} \cup \{i\}$. Or C est maximale donc $C - \{i\}$ maximale et son premier élément est u donc $C = C_{i,u} \in \mathcal{C}_m(G,i)$. CQFD

D'après cette proposition, on peut se contenter, pour calculer les cliques maximales de G, de les chercher dans $\bigcup_{1\leq i\leq n}(\mathcal{C}_m(G,i))$. D'autre part, avec la proposition3.2.3, une clique $C_{i,u}$ est maximale s'il n'existe pas j tel que $C_{i,u} \subsetneq C_{j,i}$.

Proposition 3.3.5 : S'il existe j tel que $|C_{i,u}| \leq |C_{j,i}|-1$ et $u \in C_{j,i}$ alors $C_{i,u} \subsetneq C_{j,i}$
preuve : $C_{j,i}-\{i,j\} \subset G(N^+(i))$: clique de $G(N^+(i))$ contenant u
$C_{i,u}-\{i\} \subset G(N^+(i))$: clique maximale de $G(N^+(i))$ avec u comme premier élément.
Soit $x \in C_{j,i} - \{i\}$ si $x = u$ alors $x \in C_{i,u} - \{i\}$ sinon x adjacent à i,j,u et $x > i,j,u$ alors $x \in C_{i,u} - \{i\}$. Donc $C_{j,i} - \{j\} \subset C_{i,u}$. Or $|C_{i,u}| \leq |C_{j,i}| - 1$ donc $C_{i,u} = C_{j,i} - \{j\}$ d'où $C_{i,u} \subsetneq C_{j,i}$. CQFD

Inversement, si $C_{i,u} \subsetneq C_{j,i}$ alors $u \in C_{j,i}$ et $|C_{i,u}| \leq |C_{j,i}| - 1$.
Proposition 3.3.6 : $C_{i,u}$ est maximale ssi il n'existe pas un j tel que $u \in C_{j,i}$ et $|C_{i,u}| \leq |C_{j,i}| - 1$.

Cette proposition nous permet d'élaborer un algorithme de calcul des cliques maximales de G. Notre algorithme calcule toutes les $C_{i,u}$ et parmi ces cliques, il ne garde que celles pour qui il n'existe pas de j tel que $u \in C_{j,i}$ et $|C_{i,u}| \leq |C_{j,i}| - 1$.

Cliquesmax (G, Cm)
Entrées : graphe G 2-triangulé avec le même ordre pour la 2-triangulation et la triangulation des sous-graphes induits.
Sorties : Cm ensemble des cliques maximales de G

/* Calcul des $C_{i,u}$ */
pour i=1 a n
 recherche des cliques maximales de $G(N^+[i])$
 $C_{i,u}$ = chaque clique maximale de $G(N^+[i])$ réunion i
/* Test de maximalité */
pour i=n a 1
 pour tout $C_{i,u}$
  si pour j=i-1 a 1
   pour tout $C_{j,i}$ $u \notin C_{j,i}$ ou $|C_{i,u}| > |C_{j,i}| - 1$
   alors $C_{i,u}$ est maximale

Théorème 3.3.7 : ***cliquesmax*** calcule les cliques maximales de G avec une complexité en $O(nm)$.

Nous disposons maintenant de deux algorithmes qui vont nous permettre de comparer la TR^2-Décomposition et les autres algorithmes en tenant compte du coût réel de la méthode.

4 EVALUATION EXPÉRIMENTALE

Nous avons choisi de tester nos algorithmes de résolution sur des CSP générés de manière aléatoire en tenant compte des durées d'exécution uniquement. Les algorithmes utilisés sont la TR^2-Décomposition + MCS, la TR^2-Décomposition + AB et FC, l'un des meilleurs algorithmes de résolution avec MAC. Pour la résolution des sous-problèmes générés, on utilise aussi FC. Il est à noter que les expérimentations ont été aussi lancées avec MAC, et que les résultats obtenus étaient similaires à ceux de FC. On s'est fixé sur des problèmes avec 40 variables, en faisant varier la taille des domaines, le nombre de contraintes et leur dureté. Il est nécessaire de rester assez proche du seuil de difficulté pour les problèmes générés. En effet cette méthode est inutile dans le cas où le problème est facile du fait du coût non négligeable des algorithmes de 2-triangulation et de recherche de cliques maximales, et aussi qu'il faut très peu de temps à FC pour le résoudre. Pour chaque point de cet espace des problèmes, la résolution est exécutée sur 30 instances. Pour mieux cerner le comportement des algorithmes, on distingue les problèmes consistants, des inconsistants et les durées d'exécution des différentes parties de la TR^2-Décomposition.
Dans la colonne CSP, n, d, c, t sont respectivement le nombre de variables, la taille des domaines, le nombre de contraintes et leur dureté (le pourcentage de couples interdits). SAT représente le nombre de problèmes consistants et UN, celui des inconsistants. micro est la durée d'exécution moyenne de l'algorithme de génération de la micro-structure, 2-tri celle de ***Deux-triang***, clicm, ***Cliquesmax***, et $nbcm$ est le nombre de cliques maximales. Ts est le temps minimum moyen pour résoudre un sous-problème consistant, Tu le temps moyen de résolution du sous-problème le plus difficile pour les instances inconsistantes, $Ttot$ le temps moyen de résolution de l'ensemble des sous-problèmes. Dans la colonne FC, Tm représente le temps moyen de résolution

d'un CSP.

On observe donc une grande efficacité des algorithmes pour la TR^2-Décomposition au niveau chronométrique. Au niveau de la résolution, $Ttot$ semble ne pas le critère le plus pertinent pour évaluer la TR^2-Décomposition, du fait qu'il ne tient pas compte du caractère indépendants des sous-problèmes. Il serait possible de s'arrêter dès qu'une solution est trouvée, ou même mieux faire une parallélisation des résolutions des sous-problèmes. Dans ce cas, la résolution s'arrête dès qu'une solution est trouvée sur l'ensemble des exécutions. Le temps de résolution pour les problèmes consistants se limite donc au temps du sous-problème consistant le plus facile(Ts), et pour les inconsistants à celui du plus dur(Tu), en rajoutant à cela les temps de génération de la micro-structure, de 2-triangulation et de recherche des cliques maximales. Tu et Ts témoignent mieux de la qualité de cette méthode et montrent une amélioration très significative des temps de FC. De même, comme on pouvait s'y attendre, la TR^2-Décomposition + AB donne les meilleurs résultats grâce à une meilleure décomposition avec un plus grand nombre de sous-problèmes mais de taille réduite.

CSP	***SAT***	***UN***	***TR2-Décomposition + MCS***							***FC***		
<n.d.c.t>			micro	2-tri	clicm	nbcm	Ts	Tu	Ttot	Ts	Tu	Tm
<40.10.390.20>	27	3	0	0,03	0,08	15	0,01	2,74	5,19	1,09	3,3	1,31
<40.20.234.40>	3	27	0,01	0,42	0,89	44	0	10,63	61,8	4,47	15,28	14,2
<40.20.390.25>	30	0	0,01	0,58	0,88	29	0,66	-	550,17	27,17	-	27,17
<40.20.390.30>	0	30	0,01	0,55	0,79	35	-	12,18	77,1	-	16,39	16,39
<40.30.234.45>	0	30	0,01	1,68	2,72	103	-	50,44	396,47	-	86,56	86,56
<40.30.312.35>	28	2	0,01	2,18	2,79	59	0,6	1330,67	11526,62	1427,04	1549,22	1435,18

Tableau 3 : durées d'exécution en s des algorithmes de la TR2-Décomposition + MCS et de FC

CSP	***SAT***	***UN***	***TR2-Décomposition + AB***							***FC***		
<n.d.c.t>			micro	2-tri	clicm	nbcm	Ts	Tu	Ttot	Ts	Tu	Tm
<40.10.390.20>	27	3	0	0,06	0,05	53	0	0,66	8,32	1,09	3,3	1,31
<40.20.234.40>	3	27	0,01	0,55	0,43	216	0	3,2	109,14	4,47	15,28	14,2
<40.20.390.25>	30	0	0,01	0,57	0,43	147	0	-	2388,93	27,17	-	27,17
<40.20.390.30>	0	30	0,01	0,64	0,42	179	-	1,95	71,47	-	16,39	16,39
<40.30.234.45>	0	30	0,01	2,17	1,39	350	-	11,55	778,02	-	86,56	86,56

Tableau 4 : durées d'exécution en s des algorithmes de la TR2-Décomposition + AB et de FC

micro : génération de la micro-structure
2-tri : Deuxtriang
clicm : Cliquesmax
nbcm : nombre de cliques maximales
Ts : durée sur les instances consistantes
Tu : durée sur les instances inconsistantes
Ttot : durée pour la résolution de l'ensemble des sous-problèmes
Tm : durée moyenne

5 CONCLUSION

Dans [Jégou, 1993], Jégou nous propose une nouvelle méthode de résolution de CSP, basée sur la décomposition de la micro-structure utilisant des propriétés des graphes triangulés. Dans [Chmeiss, 2003], une généralisation de cette méthode est donnée avec une généralisation des graphes triangulés, les graphes CSG^k ayant des propriétés communes. Cette méthode, malgré de bons résultats, souffrait de l'inefficacité des algorithmes de manipulations des graphes CSG^2 dont le coût prohibitif empêchait son utilisation sur des problèmes de taille satisfaisante. Nous avons proposé ici des algorithmes de 2-triangulation et de recherche de cliques maximales dans les graphes 2-triangulés qui permettent de s'affranchir de cette restriction. On a mené des expérimentations qui nous ont montré un gain très significatif d'efficacité de la TR^2-Décomposition par rapport à FC et MAC, dans le cas d'une résolution en parallèle des sous-problèmes générés. Des travaux futurs devraient permettre de trouver un ordre CSG^2 pour la 2-triangulation qui approche plus l'optimalité au niveau de la taille maximale des cliques générées et ainsi conduire à une meilleure décomposition.

BIBLIOGRAPHIE

[Jégou, 1993] Jégou P. : Decomposition of domains based on the micro-structure of finite constraint-satifaction problems. Proceedings of the Eleventh National Conference on Artificial Intelligence (1993).

[Chmeiss, 2003] Chmeiss A., Jégou P., Keddar L. : Sur une généralisation des graphes triangulés et son application pour la décomposition de domaines dans les csp. Actes JNPC (2003).

[Haralick, 1980] Haralick R. M., Elliott G. L. : Increasing tree search efficiency for constraint satisfaction problems. Artificial Intelligence, 14 (1980).

[Golumbic, 1980] Golumbic M. : Algorithmic Graph Theory and Perfect Graphs, chapter 4. Academic Press (1980).

[Fulkerson, 1965] Fulkerson, Gross : Incidence matrices and interval graphs. Pacific J. Math. 15, (1965).

[Gavril, 1972] Gavril : Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. SIAM J. Comput. 1, (1972).

[Kjaerulff, 1990] Kjaerulff : Triangulation of graphs - algorithms giving small total state space. Technical report, Judex R.R. Aalborg, Denmark (1990).

[Arnborg, 1987] Arnborg, Corneil, Proskurowski : Complexity of finding embeddings in a k-tree. SIAM J. Disc. Meth, 8 (1987).

[Berry, 1999] Berry A. : A wide-range efficient algorithm for minimal triangulation. Proceedings of SODA'99 SIAM Conference, USA (1999).

[Chmeiss, 1997] Chmeiss A., Jégou P. : A generalization of chordal graphs and the maximum clique problem. Information Processing Letters, 62 (1997).

[Habbas, 2000] Habbas, Singer, Krajecki : Domain decomposition for parallel resolution of constraints satisfaction problems with openmp. Proceedings of the second European Workshop on OpenMP, Edinburg, Scotland, UK. (2000).

[Tarjan, 1984] Tarjan, Yannakakis : Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. SIAM Journal on Computing, 13 (1984).

[Sabin, 1994] Sabin D., Freuder E.C. : Contradicting Conventional Wisdom in Constraint Satisfaction (1994). Proceedings of the Second International Workshop on Principles and Practice of Constraint Programming, PPCP'94, Rosario, Orcas Island, Washington, USA. (1994).