Inférence de cliques pour la résolution de Max-CSP

Mohand Ou Idir Khemmoudj, Hachemi Bennaceur

Inférence de Cliques pour la Résolution de Max-CSP*

Mohand Ou Idir Khemmoudj **Hachemi Bennaceur**

LIPN-CNRS UMR 7030, 99 Av J-B. Clément
93430 Villetaneuse, France
{MohandOuIdir.Khemmoudj, Hachemi.Bennaceur}@lipn.univ-paris13.fr

Résumé

Peu de travaux sur les problèmes CSP, Max-CSP et CSP valués ont été réalisés dans le domaine de l'Optimisation Combinatoire alors que ce domaine renferme de nombreux outils algorithmiques qui peuvent servir à la résolution de ces problèmes. Dans ce papier, nous décrivons une technique d'inférence d'ensembles de cliques à partir du Max-CSP que nous exploitons pour définir une nouvelle modélisation en Programmation Linéaire (PL) du Max-CSP. Nous montrons que les estimations classiques du nombre minimum de contraintes violées (bornes inférieures) basées sur la consistance d'arc telles que DAC[13], RDAC[8] et WAC[1] peuvent être traduites par des relaxations duales Lagrangiennes du PL. Nous tirons aussi profit de cette modélisation pour développer de nouvelles bornes inférieures. Dans un premier temps, afin d'avoir une idée sur la qualité des bornes inférieures obtenues à partir du PL, nous avons évalué la valeur de la relaxation continue du PL à l'aide du solveur CPLEX, puis nous avons développé une méthode du type Branch and Bound intégrant une relaxation Lagrangienne du PL. Les résultats obtenus sont très encourageants et ouvrent de nouvelles perspectives à exploiter au mieux les outils de la programmation linéaire pour contribuer à la résolution de ces problèmes.

Abstract

Few works on problems CSP, Max-CSP and weighted CSP was carried out in the field of Combinatorial Optimization, whereas this field contains many algorithmic tools which can be used for the resolution of these problems. In this paper, we propose a new modeling in linear programming (LP) of the problem Max-CSP, then we show that the traditional estimates of the minimum number of violated constraints (lower bounds) based on the arc-consistency such as DAC[13], RDAC[8] and WAC[1] can be expressed by dual Lagrangean relaxations of the LP. We benefit as from this modeling to develop new lower bounds. Initially, in order to have an idea on the quality of the lower bounds obtained from the LP, we evaluated the value of the continuous relaxation of LP using CPLEX tool, then we developed a Branch and Bound method integrating a Lagrangean relaxations of the LP. The results obtained are very encouraging and open new prospects to exploit the linear programming tools as well as possible to contribute to the resolution of these problems.

1 Introduction

Le problème de satisfaction de contraintes (CSP pour Constraint Satisfaction Problem) est défini par un ensemble de variables, ayant chacune un domaine de valeurs possibles, et par un ensemble de contraintes. Chaque contrainte limite les combinaisons de valeurs que peuvent prendre les variables sur lesquelles elle pèse. Une solution d'un CSP est une affectation satisfaisant toutes les contraintes.

Dans beaucoup d'applications, les contraintes sont antagonistes puisque la satisfaction de certaines d'entre elles se fait au détriment d'autres. Il est donc impossible de satisfaire toutes les contraintes. On peut alors être intéressé par la recherche d'une solution qui minimise le nombre de contraintes insatisfaites. Dans la littérature CSP, ce problème est référencé Max-CSP, problème de satisfaction de contraintes maximale.

Comme pour beaucoup d'autres problèmes d'Optimisation Combinatoire, l'algorithme le plus utilisé pour la résolution exacte de Max-CSP est l'algorithme de séparation et évaluation (branch and bound, B&B). L'efficacité du B&B dépend grandement de la qualité de la borne inférieure calculée en

*Ce travail est en partie soutenu par Electricité De France (EDF).

chaque noeud de l'arbre de recherche.

La plupart des méthodes efficaces de calcul de bornes inférieures se basent sur la notion de consistance locale. La principale difficulté à laquelle se heurtent ces méthodes est la prise en compte de manière globale de toutes les contraintes entre variables non encore affectées.

Dans la communauté de la Recherche Opérationnelle, d'autres bornes inférieures sont utilisées comme les bornes obtenues par la relaxation continue ou la relaxation Lagrangienne [6].

Dans le but de proposer de nouvelles bornes inférieures pour les Max-CSP binaires ou d'améliorer la qualité de certaines qui existent, ce travail porte sur l'étude de liens entre les techniques de consistance locale et les techniques de relaxations continue et Lagrangienne. On se base sur la notion de clique. Une clique étant un graphe complet, i.e, un graphe dont les sommets sont tous deux à deux adjacents. Nous nous intéressons aux cliques binaires qui expriment des incompatibilités entre les valeurs de deux variables du Max-CSP. Nous proposons une formulation linéaire pour toute clique binaire et montrons comment cette formulation peut être utilisée pour proposer une nouvelle formulation linaire équivalente au Max-CSP. Des expérimentations ont été réalisées et ont montré que la résolution par Programmation Linéaire en Nombre Entiers du système linéaire obtenu constitue une alternative intéressante pour la résolution des Max-CSP avec graphe de contraintes à densité faible.

Nous montrons aussi que les bornes inférieures obtenues par les techniques de consistances locales telles que DAC, RDAC et WAC peuvent être obtenues par la résolution d'une relaxation Lagrangienne d'un système linéaire contenant un nombre réduit de cliques. Nous proposons par la suite un processus d'inférence de cliques pour le calcul d'une meilleure borne inférieure notée CBB (CBB pour *Cilque-Based Bound*). Cette borne est intégrée dans l'algorithme de résolution PFC-MPRDAC [9] et l'algorithme obtenu est labellisé PFC-MPRDAC+CBB. Les expérimentations réalisées ont montré que PFC-MPRDAC+CBB permet d'obtenir des résultats très encourageants.

Le reste du papier est organisé comme suit : dans la section 2, on présentera quelques définitions ainsi qu'un état de l'art sur les méthodes de calcul de bornes inférieures pour les Max-CSP. Dans la section 3 on donnera la formulation linéaire proposée pour les cliques binaires et on annoncera le théorème qui nous permettra de présenter différentes manières de calcul de bornes inférieures. La section 4 présente la nouvelle formulation linéaire proposée pour les Max-CSP. Dans la section 5 on montre comment les bornes inférieures obtenues par les techniques de conssistances locales telles que DAC, RDAC et WAC peuvent être calculées par la résolution d'une simple relaxation Lagrangienne. La section 6 est consacrée à la présentation du processus d'inférence de cliques. Ensuite, nous présentons les expérimentations réalisées dans la section 7. Enfin, nous conclurons à la section 8.

2 Préliminaires

Dans cette section nous introduisons quelques définitions nécessaires pour la suite de l'article ainsi qu'un état de l'art sur les méthodes de calcul de bornes inférieures pour les Max-CSP.

2.1 Définitions

Définition 1 (Le problème de satisfaction de contraintes binaires)**.** *Un problème de satisfaction de contraintes binaires est la donnée d'un quadruplet (X,D,C,R) :*

- *X est un ensemble de variables* $\{X_1, X_2, \ldots, X_n\}$*;*
- *D est un ensemble de domaines* $\{D_1, D_2, \ldots, D_n\}$ *où chaque* D_i *représente les* d_i *valeurs possibles pour* X_i ($d_i \leq d$, *où d est la taille du plus grand domaine);*
- *C est un ensemble de e contraintes où chaque contrainte* C_{ij} *porte sur deux variables* X_i *et* X_j *de X;*
- *R est un ensemble de e relations où chaque* R_{ij} *définit l'ensemble des combinaisons de valeurs satisfaisant* C_{ij} : R_{ij} *est un sous-ensemble du produit cartésien* $D_i \times D_j$.

Définition 2 (Graphe de contraintes)**.** *On appelle graphe de contraintes d'un CSP, le graphe dont les sommets est l'ensemble des variables (X) et les arêtes celui des contraintes (C).*

Définition 3 (Graphe complémentaire de la Micro-structure du graphe de contraintes)**.** *On désignera par* $\mathcal{C}_\mu$ *le graphe complémentaire de la micro-structure du graphe de contraintes. C'est le graphe dont les sommets représentent les valeurs des variables et les arêtes les liens d'incompatibilité entre ces valeurs.*

Définition 4 (Support)**.** *Une valeur* v_k *d'une variable* X_i *est dite support pour la valeur* v_l *d'une autre variable* X_j *si et seulement si on a* $R_{ij}(v_k, v_l)$ ($(v_k, v_l) \in R_{ij}$).

Définition 5 (Clique)**.** *Dans le cadre d'un CSP, une union de sous-ensembles,* $E_{i_1} \subseteq D_{i_1}, E_{i_2} \subseteq D_{i_2}, ..., E_{i_m} \subseteq D_{i_m} : 1 \leq i_1 < i_2 < ... < i_m \leq n$,

forme une clique si et seulement si pour tout couple $(j_1, j_2) \in \{i_1, i_2, ..., i_m\}^2$ *tel que* $E_{j_1} \neq \emptyset$ *et* $E_{j_2} \neq \emptyset$ *on a* $C_{j_1 j_2} \in C$ *et* $\neg R_{j_1 j_2}(v_k, v_l)\ \forall (v_k, v_l) \in E_{j_1} \times E_{j_2}$. *C'est un sous graphe complet de* $\mathcal{C}_\mu$. *On dira qu'une clique est binaire si elle est l'union de deux sous-ensembles de deux variables du CSP.*

Pour une introduction au problème de recherche de cliques dans un graphe, le lecteur peut se référer à [3].

2.2 Bornes inférieures pour les Max-CSP

On se place dans la situation générale où on dispose d'une instanciation partielle $I_P = (X_{i_1} = v^{i_1}, ..., X_{i_m} = v^{i_m})$ et on introduit les notations suivantes :

- $P = \{i_1, i_2, ..., i_m\}, 0 \leq m \leq n$: ensemble des variables instanciées;
- $F = X \setminus P$: ensemble des variables futures;
- $C^P \subseteq C$: ensemble des contraintes portant sur des variables de P;
- $C^F \subseteq C$: ensemble des contraintes portant sur des variables de F;
- $C^{PF} = C \setminus (C^P \cup C^F)$: ensemble des contraintes portant sur une variable de P et une variable de F;
- $distance(I_P) = |\{C_{ij} \in C^P : \neg R_{ij}(v^i, v^j)\}|$: nombre de contraintes de C^P insatisfaites par l'instanciation partielle I_P;
- $ic_i^k = |\{j : \neg R_{ij}(v_k, v^j), j \in P\}|$: nombre de contraintes de C^{PF} qui seront violées si on étend l'instanciation partielle par l'affectation $X_i = v_k$. Ce nombre est appelé le compteur d'inconsistances (inconsistency count).

L'algorithme de résolution *Partial Forward Checking*[5] qui est un algorithme de séparation et évaluation utilise un minorant simple à calculer : $distance(I_P) + \sum_{i \in F} \min_{v_k \in D_i} ic_i^k$. C'est le nombre de contraintes violées dans C^P augmenté d'une évaluation du nombre de contraintes violées dans C^{PF}. [13], propose d'améliorer la qualité de ce minorant par l'ajout d'une évaluation du nombre de contraintes violées dans C^F. Il suppose un ordre fixe des variables et utilise des compteurs notés dac_i^k et appelés compteurs d'arc cohérence orientée (directed arc consistency counts ou DAC): $dac_i^k = \sum_{j>i} \min_{v_l \in D_j} (1 - R_{ij}(v_k, v_l))$. Le compteur dac_i^k est le nombre de variables $X_j \in F$ situées après X_i dans l'ordre utilisé et telles que $\neg R_{ij}(v_k, v_l), \forall v_l \in D_j$. La nouvelle borne inférieure est $distance(I_P) + \sum_{i \in F} \min_{v_k \in D_i} ic_i^k + \sum_{i \in F} \min_{v_k \in D_i} dac_i^k$. Le deuxième et troisième terme de l'expression précédente peuvent être combinés pour obtenir une meilleure borne : $distance(I_P) + \sum_{i \in F} \min_{v_k \in D_i} (ic_i^k + dac_i^k)$ [7]. [1] propose une technique pour s'en passer de l'ordre des variables. Elle consiste à associer à chaque contrainte $C_{ij} \in C^F$ deux poids λ_{ij} et λ_{ji} tels que $\lambda_{ij} + \lambda_{ji} = 1$. Elle associe aussi à chaque valeur v_k de chaque variable $X_i \in F$ une contribution wac_i^k telle que $wac_i^k = \sum_{j : C_{ij} \in C^F} \min_{v_l \in D_j} \lambda_{ji}(1 - R_{ij}(v_k, v_l))$. Les contributions wac_i^k sont par la suite utilisées pour le calcul de la borne inférieure WAC (Weighted Arc Consistency), $WAC = distance(I_P) + \sum_{i \in F} \min_{v_k \in D_i} (ic_i^k + wac_i^k)$. La méthode proposée dans [8] se base sur l'orientation des contraintes de C^F (Reversible Directed Arc Consistensy, RDAC). Chaque contrainte $C_{ij} \in C^F$ est orientée vers une de ses variables X_i ou X_j et les éventuelles absences de support sont comptabilisées sur la variable indiquée par cette orientation. Cela est équivalent à affecter (0,1) ou (1,0) au couple $(\lambda_{ij}, \lambda_{ji})$. [12] montre que le problème qui consiste à trouver une façon optimale d'orienter les contraintes de C^F est NP-difficile. Les algorithmes PFC-RDAC et PFC-MRDAC [8] optimisent l'orientation des contraintes de C^F en chaque noeud du B&B par recherche locale. [9] propose une amélioration de la borne obtenue par l'orientation des contraintes de C^F par l'ajout de contributions globales de sous-ensembles disjoints de F (Partition-Based Lower Bound). Il existe maintenant des techniques de consistances locales plus élégantes et plus puissante : AC^* [11], $FDAC^*$ [10] et $EDAC^*$ [4]. Dans ce papier nous établissons des liens entre l'inférence de cliques et les bornes inférieures classiques DAC, RDAC et WAC. Nous pensons que ces liens peuvent être aussi établis pour les techniques AC^*, $FDAC^*$ et $EDAC^*$.

3 Modèles linéaires à base de cliques pour les Max-CSP

Cette section montre comment la notion de clique binaire peut être utilisée pour le calcul de bornes inférieures pour les Max-CSP. Nous donnons tout d'abord une formulation linéaire pour les cliques binaires, puis nous annonçons un théorème qui nous permettra de présenter différentes manières de calcul de bornes inférieures.

Nous commençons la formulation par l'introduction, pour chaque variable X_i, de d_i variables binaires x_i^k, $k = 1, ..., d_i$ (d_i étant la taille du domaine D_i), telles que : $x_i^k = 1$ si $X_i = v_k$ et $x_i^k = 0$ si $X_i \neq v_k$.

Nous introduisons aussi, pour chaque contrainte C_{ij}, une variable binaire η_{ij} qui prendra la valeur 1 ou 0 selon que la contrainte est violée ou pas. Dans la suite de ce papier, nous noterons par x le vecteur à composantes x_i^k, par η le vecteur à composantes η_{ij} et par S l'ensemble des solutions du Max-CSP :

$$x = (x_i^k)_{i=\overline{1,n}}^{k=\overline{1,d_i}} \in S \Leftrightarrow \begin{cases} \sum\limits_{v_k \in D_i} x_i^k = 1 \; i = \overline{1,n} \\ x_i^k \in \{0,1\} \; i = \overline{1,n}, \; v_k \in D_i \end{cases}$$

Soit $\Gamma_{ij} = E_i \cup E_j$ une clique binaire[1]. Elle peut être formulée linéairement comme suit:

$$\psi(E_i, E_j) = \left(\sum_{v_k \in E_i} x_i^k \right) + \left(\sum_{v_l \in E_j} x_j^l \right) \leq 1 + \eta_{ij} \quad (1)$$

Cette contrainte est l'interprétation de la formule logique : $X_i = v_k \wedge X_j = v_l \Rightarrow \eta_{ij} = 1$, pour $(v_k, v_l) \in E_i \times E_j$. Elle signifie que si les variables X_i et X_j sont respectivement instanciées par des valeurs $v_k \in E_i$ et $v_l \in E_j$, alors la contrainte C_{ij} est violée. Ainsi, à tout ensemble de m cliques binaires ($m \geq 0$), $\Gamma = \{\Gamma^1_{i_1 j_1}, \Gamma^2_{i_2 j_2}, ...\Gamma^m_{i_m j_m}\}$ tel que $\Gamma^t_{i_t j_t} = E^t_{i_t} \cup E^t_{j_t} (1 \leq t \leq m)$, on peut associer le système linéaire d'optimisation suivant:

$$IP(\Gamma) \begin{cases} \min \eta.\mathbb{1}_e \\ t.q : \quad \psi(E^t_{i_t}, E^t_{j_t}) \leq 1 + \eta_{i_t j_t} \quad t = \overline{1,m} \\ \qquad\quad x \in S \end{cases}$$

où $\mathbb{1}_e$ est un vecteur de e 1, e étant le nombre de contraintes dans le Max-CSP.

Théorème 1. *Toute borne inférieure du système linéaire $IP(\Gamma)$ est une borne inférieure du Max-CSP.*

Démonstration. Il suffit de montrer que la valeur optimale du système $IP(\Gamma)$ est une borne inférieure du Max-CSP. Soit $I = (v^1, v^2, \ldots, v^n)$ une solution optimale du Max-CSP, $V(I)$ son coût (le nombre de contraintes violées) et considérons le couple de vecteurs $(\bar{x}, \bar{\eta})$ tel que:

- $\bar{x} = \left(\bar{x}_i^k\right)_{i=\overline{1,n}}^{k=\overline{1,d_i}}$: $\bar{x}_i^{v^i} = 1$ et $\bar{x}_i^k = 0, \forall v_k \in D_i - \{v^i\}$;
- $\bar{\eta} = (\bar{\eta}_{ij})_{c_{ij} \in C}$: $\bar{\eta}_{ij} = 1$ si Γ contient au moins une clique Γ^t_{ij} telle que $v^i \in E^t_i$ et $v^j \in E^t_j$, 0 sinon.

Par construction, le couple $(\bar{x}, \bar{\eta})$ est une solution du système (IP). Son coût $V(\bar{x}, \bar{\eta})$ est forcément supérieur ou égal au coût $V(IP(\Gamma))$ de la meilleure solution de $IP(\Gamma)$. Comparons maintenant $V(\bar{x}, \bar{\eta})$ avec le coût $V(I)$ de la solution I du Max-CSP. On peut vérifier que si $\bar{\eta}_{ij}$ vaut 1 alors la contrainte C_{ij} est violée par la solution I et que l'inverse est faux. En effet, $\bar{\eta}_{ij}$ ne prend un 1 que si Γ contient au moins une clique contenant à la fois v^i et v^j. Cela n'est possible, par définition de clique, que si $\neg R_{ij}(v^i, v^j)$. Par contre, il se peut bien qu'on ait $\neg R_{ij}(v^i, v^j)$ sans que Γ possède une clique contenant à la fois v^i et v^j. Cela dépend du nombre et de la façon dont elles sont construites les cliques. Par conséquent, le nombre de variables η_{ij} mises à 1 est au plus égal au nombre de contraintes violées par la solution optimale I du Max-CSP, d'où $V(I) \geq V(\bar{x}, \bar{\eta}) \geq V(IP(\Gamma))$. □

Corollaire 1. *Si pour toute contrainte $C_{ij} \in C$ et tout couple de valeurs incompatibles $(v_k, v_l) \in D_i \times D_j$, l'ensemble Γ contient au moins une clique contenant à la fois v_k et v_l alors le système $IP(\Gamma)$ est équivalent au Max-CSP.*

Dans la suite, on dira qu'un ensemble Γ est complet si le système $IP(\Gamma)$ est équivalent au Max-CSP. On dira qu'il est incomplet ou partiel dans le cas contraire.

Un des intérêts majeurs d'associer un système linéaire $IP(\Gamma)$ à un Max-CSP est de pouvoir prendre en compte de manière globale les contraintes du problème. Cependant, plusieurs questions peuvent se poser. On peut les regrouper en deux questions suivantes : (1) Comment choisir l'ensemble Γ ? (2) Comment exploiter le système linéaire $IP(\Gamma)$? On tentera de donner des éléments de réponses à ces questions dans les sections qui suivent.

4 Résolution de Max-CSP par Programmation Linéaire en Nombres Entiers

Un début de réponse possible aux questions posées à la section précédente est de dire qu'il suffit de choisir un système $IP(\Gamma)$ complet et de le résoudre par la Programmation Linéaire en Nombres Entiers (PLNE). Il reste à dire quel ensemble Γ complet choisir.

Une façon classique et simple de construire un ensemble Γ complet est d'associer à chaque couple de valeurs incompatibles $(v_k, v_l) \in D_i \times D_j$ de chaque contrainte $C_{ij} \in C$ une clique contenant uniquement les valeurs v_k et v_l. L'inconvénient de cette façon de faire réside dans le fait que l'information apportée par chaque clique est pauvre. Dans ce cas, une clique nous informe uniquement de l'incompatibilité d'un seul couple de valeurs. De plus, Γ contiendra un grand nombre de cliques, autant qu'il y a de couples de valeurs incompatibles. On présente ici une meilleure façon de construire un ensemble de cliques Γ complet.

[1] Il est à noter que l'un des sous-ensembles E_i ou E_j peut être vide, dans ce cas la clique est formée uniquement par les valeurs du sous-ensemble non vide.

À toute contrainte $C_{ij} \in C$ et tout sous-ensemble $E_i \in D_i$ on peut associer la clique $\Gamma_{ij}(E_i)$ inférée comme suit :

1. $\phi_{ij}(E_i) = \{v_l \in D_j : \neg R_{ij}(v_k, v_l)\ \forall v_k \in E_i\}$: ensemble des valeurs de X_j incompatibles avec toutes les valeurs de E_i;
2. $\phi_{ji}(\phi_{ij}(E_i)) = \{v_k \in D_i : \neg R_{ij}(v_k, v_l)\ \forall v_l \in \phi_{ij}(E_i)\}$: ensemble des valeurs de X_i incompatibles avec toutes les valeurs de $\phi_{ij}(E_i)$;
3. $\Gamma_{ij}(E_i) = \phi_{ji}(\phi_{ij}(E_i)) \cup \phi_{ij}(E_i)$: clique binaire maximale qui porte sur les variables X_i et X_j et qui contient le sous-ensemble E_i de D_i. Elle est maximale dans le sens où il n'existe pas une autre clique binaire Γ'_{ij} telle que $\Gamma'_{ij} \supset \Gamma_{ij}(E_i)$.

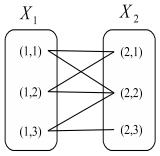

Figure 1: Exemple de contrainte

Exemple 1. *La figure 1 présente un exemple de contrainte, les arêtes représentent les incompatibilités entre valeurs. Pour cette contrainte on peut construire la clique* $\Gamma_{12}(\{(1,1)\})$ *comme suit :*

1. $\phi_{12}(\{(1,1)\}) = \{(2,1),(2,2)\}$;
2. $\phi_{21}(\phi_{12}(\{(1,1)\})) = \phi_{21}(\{(2,1),(2,2)\}) = \{(1,1),(1,2)\}$;
3. $\Gamma_{12}(\{(1,1)\}) = \{(1,1),(1,2),(2,1),(2,2)\}$.

La formulation linéaire associée à la clique $\Gamma_{12}(\{(1,1)\})$ *est :* $x_1^1 + x_1^2 + x_2^1 + x_2^2 \leq 1 + \eta_{12}$.

Remarque 1. *La clique* $\Gamma_{ij}(\{v_k\})$ *exprime au moins les incompatibilités de la valeur* v_k *de* X_i *avec toutes les valeurs de* X_j.

Théorème 2. *L'ensemble* $\Gamma_c = \{\Gamma_{ij}(\{v_k\}), \Gamma_{ji}(\{v_l\}) \mid C_{ij} \in C, v_k \in D_i, v_l \in D_j\}$ *est complet.*

Démonstration. La démonstration est évidente du fait que chaque relation R_{ij} est doublement exprimée par les ensembles de cliques $\{\Gamma_{ij}(\{v_k\}), v_k \in D_i\}$ et $\{\Gamma_{ji}(\{v_l\}), v_l \in D_j\}$. □

Les expérimentations effectuées ont montré que la résolution de $IP(\Gamma_c)$ est une alternative intéressante pour les Max-CSP avec graphe de contraintes à densité faible. Les résultats obtenus par la PLNE peuvent être améliorés par la réduction du nombre de cliques dans Γ_c.

5 Inférence de cliques pour l'expression de l'arc consistance

Dans cette section nous montrons comment les bornes inférieures telles que DAC, RDAC et WAC peuvent être obtenues par le procédé d'inférence de cliques.

Considérons l'ensemble $\Gamma^0 = \{\Gamma_{ij}(D_i), \Gamma_{ji}(D_j), C_{ij} \in C\}$ au quel est associé le système linéaire suivant

$$IP(\Gamma^0) \begin{cases} \min & \sum_{C_{ij} \in C} \eta_{ij} \\ tq: & \psi(D_i, \phi_{ij}(D_i)) \leq 1 + \eta_{ij} \quad \forall C_{ij} \in C \\ & \psi(\phi_{ji}(D_j), D_j) \leq 1 + \eta_{ij} \quad \forall C_{ij} \in C \\ & \\ & x \in S \end{cases}$$

On rappelle que $\Gamma_{ij}(D_i)$ est une clique maximale contenant toutes les valeurs de X_i ainsi que les valeurs de X_j qui n'ont pas de support sur D_i. De même, $\Gamma_{ji}(D_j)$ contient toutes les valeurs de X_j ainsi que les valeurs de X_i qui n'ont pas de support sur D_j.

Théorème 3. *La valeur* $V(IP(\Gamma^0))$ *du système* $IP(\Gamma^0)$ *est au moins égale à la meilleure borne inférieure qu'on peut obtenir par DAC, RDAC et WAC.*

Démonstration. Comme il a été montré dans [1, 2] que DAC et RDAC sont des cas particuliers de WAC, il suffit de montrer que la meilleure borne qu'on peut obtenir par WAC est un minorant de $V(IP(\Gamma^0))$.

Soit $\lambda = (\lambda_{ij}, \lambda_{ji})_{c_{ij} \in C} : \lambda_{ij} + \lambda_{ji} = 1$, un vecteur de poids qu'on peut associer aux contraintes du Max-CSP pour obtenir la meilleure borne inférieure qu'on peut calculer par WAC. À la racine de l'arbre de recherche cette borne est donnée par $\sum_{i=1}^{n} \min_{v_k \in D_i} wac_i^k$, où $wac_i^k = \sum_{j:C_{ij} \in C} \min_{v_l \in D_j} \lambda_{ji}(1 - R_{ij}(v_k, v_l))$.

Les éléments du vecteur λ peuvent aussi être utilisés comme multiplicateurs de Lagrange pour calculer une borne inférieure du système $IP(\Gamma^0)$ par relaxation Lagrangienne de toutes les contraintes exceptées les contraintes d'affectations $x \in S$. Les poids $\lambda_{ij}, \lambda_{ji}, (\forall C_{ij} \in C)$ seront associés aux contraintes linaires $\psi(D_i, \phi_{ij}(D_i)) \leq 1 + \eta_{ij}$ et $\psi(\phi_{ji}(D_j), D_j) \leq 1 + \eta_{ij}$ respectivement. La relaxation Lagrangienne de $IP(\Gamma^0)$ qu'on obtient est la suivante : $LR(\lambda, \Gamma^0) = \min_{x \in S} L(x, \lambda)$, où $L(x, \lambda)$ est la fonction de Lagrange obtenue comme suit :

$$\begin{aligned} L(x,\lambda) &= \sum_{C_{ij} \in C} (1 - \lambda_{ij} - \lambda_{ji})\eta_{ij} \\ &+ \sum_{C_{ij} \in C} \lambda_{ij} \left[\sum_{v_k \in D_i} x_i^k + \sum_{v_l \in \phi_{ij}(D_i)} x_j^l - 1 \right] \\ &+ \sum_{C_{ij} \in C} \lambda_{ji} \left[\sum_{v_k \in \phi_{ji}(D_j)} x_i^k + \sum_{v_l \in D_j} x_j^l - 1 \right] \end{aligned}$$

Après simplification, la fonction de Lagrange peut s'écrire : $L(x,\lambda) = \sum_{i=1}^{n} \sum_{v_k \in D_i} wac_i^k.x_i^k$. Le problème Lagrangien $LR(\lambda,\Gamma^0)$ est décomposable en sous-problèmes indépendants, un par variable du Max-CSP. La résolution du sous-problème relatif à une variable X_i revient à affecter un 1 à la variable x_i^k qui a la plus petite contribution wac_i^k. Par conséquent, $V(LR(\lambda,\Gamma^0)) = \sum_{i=1}^{n} \min_{v_k \in D_i} wac_i^k$, donc la meilleure borne inférieure obtenue par WAC est aussi une borne inférieure de $IP(\Gamma^0)$. □

6 Inférence de cliques pour le calcul de nouvelles bornes inférieures

Dans cette section on montre comment on peut inférer un système de cliques de taille réduite pour calculer en temps raisonnables des bornes inférieures de qualité meilleure que celles obtenues par DAC, RDAC et WAC.

La démarche consiste à inférer itérativement une suite de p ensembles de cliques $\Gamma^1, \Gamma^2, ..., \Gamma^p$. Chacun de ces ensembles se compose d'autant de cliques binaires qu'il y a de contraintes portant sur des variables futures. Ils sont construits de telle façon que $V(LR(\mathbb{1},\Gamma^1)) < V(LR(\mathbb{1},\Gamma^2)) < V(LR(\mathbb{1},\Gamma^3)) < ... < V(LR(\mathbb{1},\Gamma^{p-1})) = V(LR(\mathbb{1},\Gamma^p))$, où $V(LR(\mathbb{1},\Gamma^q))$ $(1 \leq q \leq p)$ est la valeur de la relaxation Lagrangienne qu'on obtient par la dualisation de toutes les contraintes de $IP(\Gamma^q)$ avec des multiplicateurs de Lagrange tous égaux à 1.

Le processus d'inférence commence par le calcul d'un ensemble Γ^1 associant à chaque contrainte C_{ij} une des deux cliques $\Gamma_{ij}(D_i)$ ou $\Gamma_{ji}(D_j)$. Pour cela, le problème dual Lagrangien $\max_{\lambda} V(LR(\lambda,\Gamma^0))$ introduit à la section précédente est résolu de manière approchée par la même méthode de recherche locale (**GreedyOpt**) utilisée par l'algorithme MRDAC pour diriger le graphe de contraintes.

GreedyOpt

1. STOP $\leftarrow$ *false*, $\forall C_{ij} \in C, \lambda_{ij} = 1, \lambda_{ji} = 0$
2. **WHILE**(! STOP)
   (a) STOP $\leftarrow$ *true*
   (b) $\forall C_{ij} \in C$
      i. $\lambda' \leftarrow \lambda, \lambda'_{ij} \leftarrow \lambda_{ji}, \lambda'_{ji} \leftarrow \lambda_{ij}$
      ii. **if** $(\min_{x \in S} L(x,\lambda') > \min_{x \in S} L(x,\lambda))$ **then** $\{\lambda \leftarrow \lambda'$, STOP $\leftarrow$ *false*$\}$

Le résultat de la procédure **GreedyOpt** est une affectation de (1,0) ou (0,1) à chaque couple $(\lambda_{ij}, \lambda_{ji})$. Γ^1 est alors initialisé en associant à chaque contrainte C_{ij} la clique $\Gamma_{ij}(D_i)$ si $\lambda_{ij} = 1$ ou la clique $\Gamma_{ji}(D_j)$ sinon. On vérifiera que la borne inférieure qu'on obtient après l'exécution de **GreedyOpt** est $V(LR(\mathbb{1},\Gamma^1)) = \sum_{i=1}^{n} \min_{v_k \in D_i} cc_i^k - e$, où cc_i^k est le nombre de cliques dans Γ^1 contenant la valeur v_k de la variable X_i. À ce niveau $V(LR(\mathbb{1},\Gamma^1))$ correspond exactement à la borne inférieure calculée par RDAC. La suite du processus vise à améliorer la qualité de cette borne. On appellera cc_i^k le compteur de cliques associé à la valeur v_k de la variable X_i.

Considérons les notations suivantes :

- $\Gamma^q = \{\Gamma_{ij}^q, C_{ij} \in C\}$: ensemble de cliques générées à l'itération q du processus;
- $\Gamma_{ij}^q = E_{ij}^q \cup E_{ji}^q$: clique associée à la contrainte C_{ij} à l'itération q du processus. E_{ij}^q et E_{ji}^q sont respectivement des sous ensembles de D_i et D_j;
- $min_i = \min_{v_k \in D_i} cc_i^k$: le minimum des compteurs de cliques associés aux valeurs de X_i;
- $rc_i^k = cc_i^k - min_i$: coût réduit de la valeur v_k de X_i.
- $MIN(E_{ij}^q) = \{v_k \in E_{ij}^q : rc_i^k = 0\}$: ensemble des valeurs de $E_{ij}^q \subseteq D_i$ apparaissant un nombre de fois minimum dans les cliques de Γ^q.
- $MIN(E_{ji}^q) = \{v_l \in E_{ji}^q : rc_j^l = 0\}$: ensemble des valeurs de $E_{ji}^q \subseteq D_j$ apparaissant un nombre de fois minimum dans les cliques de Γ^q.

Théorème 4. *Soit $\Gamma^{q'}$ l'ensemble de cliques obtenu en remplaçant dans Γ^q la clique $\Gamma_{ij}^q = E_{ij}^q \cup E_{ji}^q$ par la nouvelle clique $\Gamma_{ij}^{q'} = \Gamma_{ij}(MIN(E_{ij}^q))$ (resp. par $\Gamma_{ij}^{q'} = \Gamma_{ji}(MIN(E_{ji}^q))$). La valeur $V(LR(\mathbb{1},\Gamma^{q'}))$ est au moins égale à $V(LR(\mathbb{1},\Gamma^q))$.*

Démonstration. On donne la démonstration seulement pour le cas où $\Gamma_{ij}^q = E_{ij}^q \cup E_{ji}^q$ est remplacée par $\Gamma_{ij}^{q'} = \Gamma_{ij}(MIN(E_{ij}^q)) = \phi_{ji}(\phi_{ij}(MIN(E_{ij}^q))) \cup \phi_{ij}(MIN(E_{ij}^q))$, la démonstration de l'autre cas se fait de manière analogue.

Comme il y a seulement la clique associée à la contrainte C_{ij} qui est remplacée par une nouvelle clique, les compteurs de cliques associés aux valeurs des variables autres que X_i et X_j restent inchangées. Les contributions de ces variables dans le calcul de la borne inférieure restent les mêmes. Comptons maintenant les nouvelles contributions de X_i et X_j. Parmi les valeurs de D_i, seules celles qui apparaissaient dans la clique

Γ^q_{ij} (les valeurs de E^q_{ij}) et qui n'apparaissent pas dans la clique $\Gamma^{q'}_{ij}$ peuvent voir leur compteurs de cliques diminuer. Comme $\Gamma_{ij}(MIN(E^q_{ij}))$ est une clique maximale qui contient toutes les valeurs de l'ensemble $MIN(E^q_{ij})$, aucune valeur de $MIN(D_i)$ ne verra alors son compteur de cliques diminuer. Par conséquent, la contribution de X_i reste inchangée. D'autre part, toutes les valeurs de D_j qui apparaissaient dans la clique Γ^q_{ij} (les valeurs de E^q_{ji}) apparaissent dans la clique $\Gamma^{q'}_{ij}$, car les valeurs de $MIN(E^q_{ij})$ et de E^q_{ji} sont toutes deux à deux incompatibles puisqu'elles sont toutes contenues dans la clique Γ^q_{ij}. En plus des valeurs E^q_{ji}, la clique $\Gamma^{q'}_{ij}$ contient toutes les valeurs de X_j qui ne sont pas dans E^q_{ji} et qui sont incompatibles avec les valeurs de $MIN(E^q_{ij})$. On peut en déduire que le nombre de cliques de $\Gamma^{q'}$ où une valeur $v_l \in D_j$ apparaît est supérieur ou égal au nombre de cliques de Γ^q où cette même valeur v_l apparaît. La nouvelle contribution de la variable X_j est donc forcément supérieure ou égale à l'ancienne, $V(LR(\mathbb{1}, \Gamma^{q'})) \geq V(LR(\mathbb{1}, \Gamma^q))$. □

Il est clair d'après la démonstration du théorème précédent qu'on peut annoncer le corollaire suivant :

Corollaire 2. *Si* $MIN(E^q_{ji}) = \emptyset$ *et* $\forall (v_k, v_l) \in MIN(E^q_{ij}) \times MIN(D_j)$ *on a* $\neg R_{ij}(v_k, v_l)$ *alors si on remplace dans* Γ^q *la clique* Γ^q_{ij} *par* $\Gamma_{ij}(MIN(E^q_{ij}))$ *on obtient un nouvel ensemble de cliques* $\Gamma^{q'}$ *tel que* $V(LR(\mathbb{1}, \Gamma^{q'})) = V(LR(\mathbb{1}, \Gamma^q)) + 1$.

Le processus continue par la génération des ensembles $\Gamma^2, \Gamma^3, ..., \Gamma^p$. Chaque ensemble $\Gamma^q, 2 \leq q \leq p$ est obtenu à partir de Γ^{q-1} par la procédure **Descente**. L'étape 2 de la procédure **Descente** consiste

Descente

1. $\Gamma^q \leftarrow \Gamma^{q-1}$
2. $\forall C_{ij} \in C$ **if** $(\lambda_{ij} = 1)$
   then $\Gamma^q \leftarrow \Gamma^q \setminus \Gamma^q_{ij} \cup \Gamma_{ij}(MIN(E^{q-1}_{ij}))$
   else $\Gamma^q \leftarrow \Gamma^q \setminus \Gamma^q_{ij} \cup \Gamma_{ji}(MIN(E^{q-1}_{ji}))$
3. $NewLB = V(LR(\mathbb{1}, \Gamma^q))$

à remplacer itérativement dans Γ^{q-1} chaque clique Γ^{q-1}_{ij} par une des deux cliques $\Gamma_{ij}(MIN(E^{q-1}_{ij}))$ ou $\Gamma_{ji}(MIN(E^{q-1}_{ji}))$ selon que $(\lambda_{ij}, \lambda_{ji}) = (1,0)$ ou (0,1). À chaque remplacement, les compteurs de cliques de deux variables peuvent être modifiés. Ces modifications peuvent donner lieu à l'augmentation de la borne inférieure si on est dans le cas favorable du corollaire 2. Le nombre de tels remplacements favorables est la quantité par laquelle la borne inférieure est augmentée après la fin de l'exécution de la procédure **Descente**. Si cette augmentation est positive, le processus continue par un nouvel appel à la procédure **Descente** pour générer l'ensemble Γ^{q+1}, sinon le processus est arrêté. Le paramètre p est alors égal au nombre de fois que la procédure **Descente** est exécutée plus 1.

Remarque 2. *La quantité par laquelle la borne inférieure est augmentée après la fin du processus dépend de l'ordre dans lequel les contraintes sont considérées lors de chaque appel à la procédure* ***Descente****. Nous avons constaté expérimentalement qu'il est préférable de traiter en priorité les contraintes favorables à l'augmentation de la borne inférieure (voir corollaire 2). La procédure* ***Descente*** *est alors implémentée de manière à ce que, lors du premier appel, les contraintes favorables à l'augmentation de la borne inférieure soient traitées en priorité.*

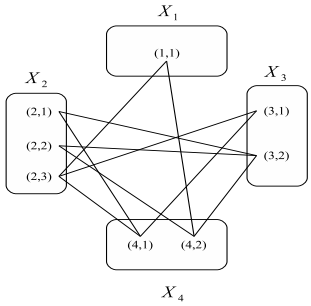

Figure 2: Exemple de Max-CSP

Exemple 2. *La figure 2 présente un exemple de Max-CSP contenant 4 variables* $(X = \{X_1, X_2, X_3, X_4\})$ *et 5 contraintes* $(C = \{C_{12}, C_{14}, C_{23}, C_{24}, C_{34}\})$. *Les arcs sur la figure représentent les couples de valeurs incompatibles. En partant du système de cliques* Γ^0 *associé et en résolvant par la procédure* ***Greedy-Opt*** *le problème dual laganragien* $\max_{\lambda} V(LR(\lambda, x))$ *on obtient le vecteur* $\lambda = (1,0,1,0,1,0,1,0,1,0)$. *L'ensemble* Γ^1 *est alors initialisé comme suit :*
$\Gamma^1 = \{\Gamma_{12}(D_1), \Gamma_{14}(D_1), \Gamma_{23}(D_2), \Gamma_{24}(D_2), \Gamma_{34}(D_3)\}$
$= \{\{(1,1),(2,3)\}, \{(1,1),(4,2)\}, \{(2,1),(2,2),(2,3)\}, \{(2,1),(2,2),(2,3)\}, \{(3,1),(3,2)\}\}$.

Le compteur de cliques associé à chaque valeur (i,k) *est le nombre de cliques dans* Γ^1 *où elle apparaît. Ces compteurs sont donnés par le tableau suivant :*

$i\backslash k$	1	2	3	min_i	$i\backslash k$	1	2	3	min_i
1	2	♯	♯	2	3	1	1	♯	1
2	2	2	3	2	4	0	1	♯	0

La borne inférieure est donc $V(LR(\mathbb{1}, \Gamma^1)) = \sum_{i=1}^{n} \min_i - e = 2 + 2 + 1 - 5 = 0$. *Le processus*

continue d'abord par repérer les contraintes favorables à l'augmentation de la borne inférieure pour qu'elles soient considérées les premières lors du premier appel à ***Descente*** *(voir remarque 4). Ici aucune contrainte n'est favorable à l'augmentation de la borne inférieure. Elles sont donc considérées dans l'ordre* $C_{12}, C_{14}, C_{23}, C_{24}, C_{34}$. *L'ensemble* Γ^2 *qu'on obtient après le premier passage de* ***Descente*** *est* $\Gamma^2 = \{\{(1,1),(2,3)\},\{(1,1),(4,2)\},\{(2,1),(2,2),(3,2)\},\{(2,1),(2,2),(2,3)\},\{(3,1),(4,1)\}\}$. *Seules les cliques associées à* C_{23} *et* C_{34} *sont remplacées par de nouvelles cliques. Il y a d'abord* $\Gamma_{23}(D_2)$ *qui est remplacée par* $\Gamma_{23}(MIN(E^1_{23})) = \Gamma_{23}(\{(1,1),(1,2)\}) = \{(2,1),(2,2),(3,2)\}$. *Ce remplacement donne lieu à la modification des compteurs de cliques associés aux valeurs des variables* X_2 *et* X_3, cc^3_2 *diminue de 1 (*$cc^3_2 = 2$ *) et* cc^2_3 *augmente de 1 (*$cc^2_3 = 2$*). À ce niveau,* $MIN(E^1_{34}) = \{(3,1)\}$. *Suite à ce premier remplacement, il y a le traitement de la clique associée à* C_{23} *qui ne donne aucune modification, puis* $\Gamma_{34}(D_3)$ *est remplacée par* $\Gamma_{34}(MIN(E^1_{34})) = \Gamma_{34}(\{(3,1)\}) = \{(3,1),(4,1)\}$. *Le nouveau tableau des compteurs de cliques qu'on obtient après le premier passage de* ***Descente*** *est :*

$i\backslash k$	1	2	3	min_i	$i\backslash k$	1	2	3	min_i
1	2	♯	♯	2	3	1	1	♯	1
2	2	2	2	2	4	1	1	♯	1

La nouvelle borne inférieure est $V(LR(\mathbb{1},\Gamma^2)) = \sum_{i=1}^{n} \min_i - e = 2+2+1+1-5 = 1$.

À Γ^2 *correspond le système suivant :*

$$IP(\Gamma^2)\begin{cases} min & \eta_{12}+\eta_{14}+\eta_{23}+\eta_{24}+\eta_{34} & \\ t.q: & x^1_1+x^3_2 & \leq 1+\eta_{12} \\ & x^1_1+x^2_4 & \leq 1+\eta_{14} \\ & x^1_2+x^2_2+x^2_3 & \leq 1+\eta_{23} \\ & x^1_2+x^2_2+x^3_2 & \leq 1+\eta_{24} \\ & x^1_3+x^1_4 & \leq 1+\eta_{34} \\ & x \in S & \end{cases}$$

Le premier appel à ***Descente*** *a donné lieu à l'augmentation de la borne inférieure d'une unité. Ce sont les 5 cliques construites qui ont contribué ensemble à cette augmentation. Le processus continue par un nouvel appel à* ***Descente*** *pour générer* Γ^3. *Cette fois la borne ne sera pas augmentée et le processus est alors arrêté. La borne inférieure calculée est 1.*

Remarque 3. *La complexité temporelle du processus est en* $O((e.d)^2)$*: la méthode* ***Descente*** *est exécutée au plus e fois puisque la borne inférieure initiale est au plus augmentée par e. À chaque exécution les e contraintes sont considérées et le traitement de chaque contrainte coûte* d^2. *En pratique, plus le nombre d'exécution de la méthode* ***Descente*** *croît, plus l'efficacité du processus augmente.*

La complexité spatiale du processus est en $O(e.d)$*: on a besoin d'une matrice de taille* $e \times 2d$ *pour représenter les cliques associées aux e contraintes.*

Le processus d'inférence de cliques est intégré dans la méthode de résolution PFC-MPRDAC. On dénote par PFC-MPRDAC+CBB l'algorithme résultant (CBB pour *Clique Based-Bound*). En chaque noeud de l'arbre de recherche PFC-MPRDAC effectue tous le travail nécessaire pour diriger le graphe de contraintes, donc pour résoudre approximativement le problème dual Lagrangien $\max_{\lambda} V(LR(\lambda,\Gamma^0))$. Le résultat est récupéré pour initialisé Γ^1. PFC-MPRDAC procède aussi à la partition de l'ensemble des variables futures en sous-ensembles disjoints. Chaque sous-ensemble est composé soit d'une ou de deux variables. Les contraintes qui portent sur les variables contenues dans des sous-ensembles de taille 2 correspondent aux contraintes favorables à l'augmentation de la borne inférieure. Ce sont les contraintes qui sont traitées les premières lors du premier passage de **Descente**.

Le processus d'inférence de cliques est présenté pour le cas où aucune variable n'est encore instanciée. Aux noeuds internes de l'arbre de recherche la borne inférieure calculée par PFC-MPRDAC+CBB est $CBB = distance(I_P) - |C^F| + \sum_{i \in F} \min_{v_k \in D_i}(ic^k_i + cc^k_i)$, où I_P est une instanciation partielle, F est l'ensemble des variables futures, C^F est l'ensemble des contraintes portant sur des variables de F et ic^k_i est le compteur d'inconsistances associé (inconsistensy count) à la valeur de la variable $X_i \in F$.

Les quantités $ic^k_i + cc^k_i$ sont exploitées par PFC-MPRDAC+CBB dans le cadre du filtrage. À chaque fois que les compteurs de cliques d'une variable X_i sont modifiés, les coûts réduits $rc^k_i = ic^k_i + cc^k_i - min_i \; \forall v_k \in D_i$ sont calculés. Si $rc^k_i \geq UB - CBB$ la valeur v_k est éliminée du domaine D_i, UB étant la valeur de la meilleure solution courante.

7 Expérimentations

Cette section présente les expérimentations réalisées. Les tests ont été effectués sur trois classes de Max-CSP. Chaque classe est représentée par un quadruplet $\langle n,d,e,t\rangle$ tel que n est le nombre de variables du Max-CSP, d la taille des domaines et e le nombre de contraintes. Le paramètre t est variable. Il est égal au nombre de couples de valeurs interdits par chaque contrainte (tightness). Les trois classes en question sont
1.$\langle 10,10,45,t\rangle$ 2.$\langle 15,10,50,t\rangle$ 3.$\langle 40,5,55,t\rangle$.
Les graphes de contraintes des Max-CSP appartenant

aux classes (1) sont complets. La densité des Max-CSP appartenant aux classes (2) est moyenne, tandis que celle des Max-CSP de la classe (3) est faible.

Les premières expérimentations ont porté sur l'évaluation des différentes bornes inférieures suivantes:

1. $V(LP(\Gamma_c))$: valeur de la relaxation continue du système complet $V(IP(\Gamma_c))$ (voir section 4);
2. $V(IP(\Gamma^0))$: valeur exacte du système partiel $IP(\Gamma^0)$ (voir section 5);
3. $V(LP(\Gamma^0))$: valeur de la relaxation continue du système partiel $IP(\Gamma^0)$ (voir section 5);
4. CBB : borne inférieure calculée par PFC-MPRDAC+CBB à la racine de l'arbre de recherche (voir section 6).

Pour chaque classe et pour chaque valeur de t 50 instances de Max-CSP ont été générées aléatoirement. L'outil utilisé pour le calcul des trois première bornes est le solveur CPLEX. Les résultats de l'évaluation des différentes bornes inférieures sont donnés par la figure 3. Les courbes sur la première colonne de la figure représentent les valeurs des différentes bornes inférieures. Les courbes de la deuxième colonne représentent le temps nécessaire pour le calcul de ces différentes bornes.

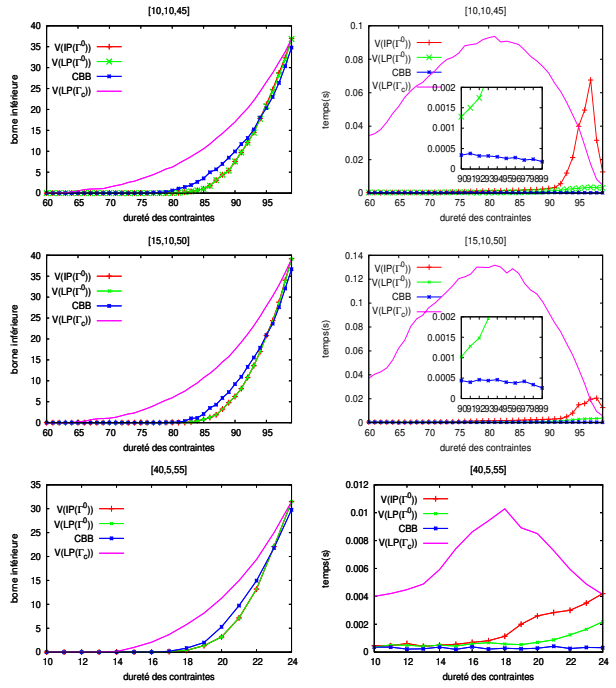

Figure 3: Comparaison des différentes bornes inférieures

On remarque que $V(LP(\Gamma_c))$ est largement supérieure aux autres bornes et que le temps nécessaire pour le calcul de $V(LP(\Gamma_c))$ est beaucoup trop élevé sauf pour la classe 3. La valeur de la relaxation continue $V(LP(\Gamma^0))$ est de bonne qualité, elle coïncide avec la valeur $V(IP(\Gamma^0))$. La valeur de CBB est meilleure que celle de $V(IP(\Gamma^0))$ sauf lorsque la dureté des contraintes est proche de 100 %. De plus CBB est obtenue dans des temps beaucoup moins élevés que ceux que nécessite la calcul de $V(LP(\Gamma^0))$.

La résolution exacte de $IP(\Gamma_c))$ est aussi évaluée. La table 1 présente les résultats obtenus pour les problèmes difficiles des trois classes[2] (t est choisi proche de la région difficile, voir la figure 4). La deuxième colonne de la table (GAP) représente l'écart entre la valeur de la relaxation continue de $IP(\Gamma_c)$ et sa valeur exacte. La troisième colonne donne le nombre de noeuds explorés en moyenne pour résoudre les problèmes générés et la quatrième colonne donne le temps de résolution en secondes. Ces résultats montrent que la PLNE est une alternative intéressante pour la résolution de Max-CSP avec graphe de contraintes à densité faible. En effet, CPLEX arrive à résoudre les problèmes de la classe $\langle 40, 5, 55, 20\rangle$ en moins de 25 millisecondes en moyenne. Ces résultats peuvent être expliqués par la qualité de la borne inférieure $V(LP(\Gamma_c))$ (l'écart entre la borne inférieure $V(LP(\Gamma_c))$ et la valeur exacte $V(IP(\Gamma_c))$ est pratiquement nul). Pour les Max-CSP des autres classes, le temps que nécessite la résolution exacte par CPLEX est élevé par rapport au temps que nécessite la méthode de résolution PFC-MRDAC. CPLEX met plus de 30 secondes pour résoudre les problèmes des classes $\langle 10, 10, 45, 87\rangle$, $\langle 15, 10, 50, 85\rangle$. Ces résultats peuvent être expliqués par le fait que la résolution de la relaxation continue prend beaucoup de temps (voir figure 3).

Table 1: Résultats de CPLEX

$\langle n, d, e, t\rangle$	GAP	noeuds	temps (s)
$\langle 10, 10, 45, 87\rangle$	9.66	2152.4	33.23
$\langle 15, 10, 50, 85\rangle$	9.42	5118.76	110.82
$\langle 40, 5, 55, 20\rangle$	0.32	0.04	0.025

Les autres expérimentations ont porté sur l'évaluation de PFC-MPRDAC+CBB. Les résultats obtenus sont donnés par la figure 4. Les courbes sur la première colonne de la figure représentent le nombre de noeuds explorés en moyenne par les trois méthodes PFC-MRDAC, PFC-MPRDAC et PFC-

[2] La résolution de 50 problèmes pour chaque valeur de t et pour chaque classe prend plusieurs jours.

MPRDAC+CBB pour résoudre les problèmes génerés. Les courbes de la deuxième colonne représentent le temps de calcul moyen qu'ont necessité les trois algorithmes pour résoudre les problèmes générés. On remarque que PFC-MPRDAC+CBB développe

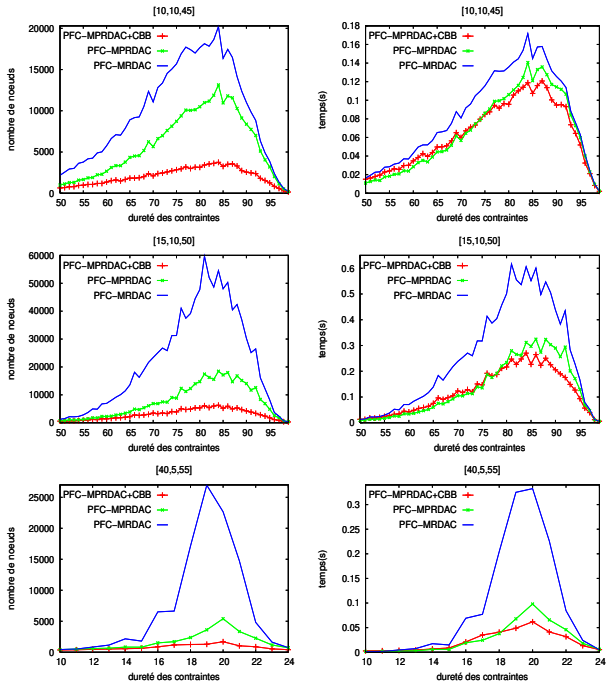

Figure 4: Comparaison de PFC-MRDAC, PFC-MPRDAC et PFC-MPRDAC+CBB

beaucoup moins de noeuds et met moins de temps que les autres algorithmes pour résoudre les instances difficiles de Max-CSP.

8 Conclusion

Dans ce papier, nous avons montré que la notion de clique peut être bien exploitée dans le cadre des Max-CSP. Nous avons proposé une formulation linéaire pour les cliques binaires et montré que cette formulation peut être utilisée pour proposer des modèles linéaires utiles pour le calcul de bornes inférieures ou la résolution exacte des Max-CSP. Nous avons proposé un algorithme d'inférence de cliques permettant d'obtenir un système linéaire facile à évaluer. Son évaluation fournit de bonnes bornes inférieures. Le système d'inférence est intégéré dans la méthode de résolution PFC-MPRDAC et l'algorithme résultant a été labellisé PFC-MPRDAC+CBB (CBB pour Clique-Based Bound). Les expérimentations réalisées ont montré que PFC-MPRDAC+CBB est plus performant que PFC-MRDAC et PFC-MPRDAC.

La sophistication de notre algorithme et son extension aux CSP valués sont nos perspectives.

References

[1] Affane, M.S., Bennaceur, H.: A Weighted Arc Consistency Technique for MAX-CSP. *ECAI* (1998) 209-213.

[2] Affane, M.S., Bennaceur, H. Schiex, T.: Comparaison de deux minorants paramétriques pour Max-CSP. *JNPC* (1999) 95-102.

[3] Bomze, I., Budinich, M., Pardalos, P., Pelillo, M.: The Maximum Clique Problem. *Handbook of Combinatorial Optimization*, 4, (1999).

[4] de Givry, S., Zytnicki, M., Heras, F., Larrosa, J.: Existential arc consistency: Getting closer to full arc consistency in weighted csps. *IJCAI* (2005) 84-89.

[5] Freuder, E., Wallace, R.: Partial Constraint Satisfaction. *Artificial Intelligence* 58, (1992) 21-70.

[6] Geoffrion, A.M.: The Lagrangean Relaxation for Integer Programming. Mathematical Programming. **2** (1974) 82-114.

[7] Larrosa, J., Meseguer, P.: Exploiting the use of DAC in Max-CSP. *CP* (1996) 308-322.

[8] Larrosa, J., Meseguer, P., Schiex, T. Maintening Reversible DAC for Max-CSP. *Artificial Intelligence* 107(1), (1999) 149-163.

[9] Larrosa, J., Meseguer, P.: Partition-Based Lower Bound for Max-CSP. *CP* (1999) 303-315.

[10] Larrosa, J., Schiex, T.: In the quest of the best form of local consistency for weighted CSP. *IJCAI* (2003) 239-244.

[11] Larrosa, J., Schiex, T. Solving Weighted CSP by Maintaining arc consistency. *Artificial Intelligence* 159(1-2), (2004) 1-26.

[12] Schiex, T.: Maximizing the reversible DAC lower bound in Max-CSP is NP-hard. Rapport technique. INRA (1998).

[13] Wallace, R.: Directed arc consistency preprocessing. Dans Meyer,M., ed., *Selected papers from the ECAI Workshop on Constraint Processing,* 923 dans LNCS. Berlin: Springer, (1994) 121-137.