Recouvrement de problèmes par des hypergraphes acycliques : analyses théorique et expérimentale

Philippe Jégou **Samba Ndojh Ndiaye** **Cyril Terrioux**

LSIS - UMR CNRS 6168
Université Paul Cézanne (Aix-Marseille 3)
Avenue Escadrille Normandie-Niemen
13397 Marseille Cedex 20
{philippe.jegou, samba-ndojh.ndiaye, cyril.terrioux}@univ-cezanne.fr

Résumé

Cette contribution s'intéresse à la notion de recouvrement de problèmes (au sens des CSPs) par des hypergraphes acycliques (ou hyper-arbres). Elle introduit une méthode de résolution fondée sur l'exploitation d'ensemble d'hypergraphes acycliques recouvrants. Ces recouvrements peuvent être assimilés à une forme d'extension de la notion classique de décomposition arborescente de réseau de contraintes. Nous étudions ici les propriétés et les relations de ces recouvrements, puis nous évaluons leur intérêt théorique pour le cas de problèmes structurés. Nous montrons que cette approche rend possible une gestion dynamique de la structure des CSPs pendant la résolution, et facilite ainsi une exploitation aisée des heuristiques dynamiques d'ordonnancement des variables. De plus, nous proposons un résultat de complexité qui améliore significativement ceux fournis précédemment dans la littérature. Enfin, nous présentons des résultats expérimentaux qui donnent une idée de l'intérêt de cette nouvelle approche sur le plan pratique.

1 Introduction

Dans de nombreux domaines de l'Intelligence Artificielle, il a été montré que les jeux de données, les informations, ou encore les connaissances représentées, possèdent généralement des propriétés structurelles. Quand ces données expriment un problème, ou bien s'il s'agit de réaliser certaines formes de raisonnement dessus, la connaissance et l'exploitation de ces propriétés structurelles s'avèrent alors cruciales. Par le passé, dans différents domaines ou formalismes, cet intérêt a souvent été vérifié : pour le test de satisfiabilité dans SAT [8, 12, 14], pour la résolution de CSPs [6], dans les réseaux Bayesiens ou probabilistes [2, 4], dans les bases de données relationnelles [1, 7], ou encore dans l'optimisation sous contraintes [3, 19]. Des résultats de complexité basés sur des propriétés topologiques de ces réseaux ont été proposés. Généralement, ces résultats sont relatifs aux caractéristiques d'une décomposition arborescente [15] du réseau considéré, décomposition que l'on peut assimiler à un hypergraphe acyclique (un hyper-arbre) recouvrant le réseau. La complexité en temps des meilleures méthodes structurelles est $O(exp(w+1))$, où w est la tree-width de la décomposition arborescente. En effet cette complexité peut considérablement améliorer la complexité de base, à savoir $O(exp(n))$ avec $w < n$, et parfois $w \ll n$. Cependant, bien que de nombreuses méthodes et plusieurs résultats théoriques aient été proposés, les avantages pratiques de ce type d'approche n'ont pas été prouvés, exceptés dans quelques travaux récents, notamment autour des CSPs Valués [3, 11, 13]. Ceci est dû, essentiellement, au fait que les bornes de complexité sont souvent obtenues au détriment de l'efficacité pratique. A l'opposé, quelques méthodes exploitant la structure des problèmes, basées sur des heuristiques qui ne garantissent pas de bornes de complexité de qualité, ont montré leur intérêt pratique [12].

Dans cette contribution, nous proposons un compromis entre la vérification de bornes de complexité théorique de qualité et la nécessité péremptoire qu'il y a à exploiter des heuristiques efficaces. De ce point de

vue, le travail présenté ici peut être considéré comme une extension de travaux récemment présentés dans le cadre du "AND/OR Branch-and-Bound" pour l'optimisation sous contraintes [13], ou dans le cadre des méthodes fondées sur la décomposition arborescente [9]. Notre approche qui s'appuie sur la notion de décomposition arborescente, constitue de fait une extension de ces derniers travaux. Toutefois, nous proposons un cadre différent conduisant à de meilleurs résultats théoriques et à une validation pratique de cette approche. En fait, nous préférons à la notion de décomposition arborescente, le concept plus général de recouvrement par hypergraphe acyclique. Etant donné un graphe $G = (X, C)$ issu de la représentation graphique du problème traité, nous considérons un recouvrement de ce graphe par un hypergraphe acyclique $H = (X, E)$: l'ensemble des sommets est le même alors que, pour toute arête $\{x, y\} \in C$, il existe une (hyper)arête $E_i \in E$ recouvrant $\{x, y\}$ ($\{x, y\} \subset E_i$). Maintenant, nous pouvons définir différentes classes d'hypergraphes acycliques qui recouvrent H. Ces classes sont définies sur la base de critères relatifs à la nature des recouvrements et des relations existant avec les méthodes de résolution : bornes issues de paramètres du type de la tree-width, préservation des séparateurs de H, fusion d'hyperarêtes voisines, capacité à l'implémentation d'heuristiques efficaces (essentiellement dynamiques).

Dans un premier temps, ces recouvrements sont étudiés théoriquement, de façon à déterminer leurs caractéristiques et leurs propriétés. Après cela, nous montrons qu'ils préservent les résultats de complexité connus, mais permettent en outre d'en améliorer certains. De plus, nous indiquons comment cette notion de recouvrement permet d'offrir un cadre naturel pour la gestion dynamique de la structure. Ainsi, pendant la résolution, nous pouvons prendre en compte, non seulement un recouvrement par hypergraphe acyclique, mais aussi un ensemble de recouvrements. Cette démarche a pour objet essentiel de faciliter une gestion dynamique des heuristiques. Grâce à ce cadre formel, nous présentons un nouvel algorithme pour lequel il est assez facile d'étendre les heuristiques connues. Par exemple, pour l'ordonnancement dynamique des variables, il devient alors possible d'agrandir dynamiquement à Δ variables supplémentaires, l'étendue du choix généralement offert et qui porte sur seulement $w + 1$ variables. Alors que dans [9] la complexité associée à ce type de démarche était en $O(exp(2(w + \Delta + 1) - s^-))$ où s^- est la taille minimum des séparateurs, nous montrons que la complexité en temps est limitée ici à $O(exp(w + \Delta + 2))$. Enfin, nous montrons comment l'implémentation est simplifiée, nous permettant de mieux apprécier l'intérêt pratique de cette approche.

Pour faciliter sa présentation, ce travail sera présenté dans le cadre des CSPs. Ceci ne restreint pas la généralité du propos, dans la mesure où l'objet de la majorité des travaux réalisés sur ce type de questions y a été développé. C'est d'autant moins gênant qu'il serait facile d'étendre ce travail à d'autres formalismes comme par exemple SAT, MAX-SAT, MAX-CSP, VCSP, ou encore les réseaux probabilistes.

Dans la seconde partie de cet article, nous introduisons différentes classes de recouvrements de graphes par des hypergraphes acycliques et nous étudions leurs relations. La troisième partie montre comment ces classes peuvent être exploitées sur le plan algorithmique. Elle fournit en outre les bornes de complexité qu'elles garantissent. Finalement, nous présentons une analyse expérimentale avant de conclure.

2 Recouvrements par hypergraphes acycliques

Un *problème de satisfaction de contraintes* (CSP) est défini par un triplet (X, D, C). X est un ensemble de n variables qui doivent être affectées dans des domaines finis de valeurs définis dans D, ceci en satisfaisant un ensemble C de contraintes. Une solution est une affectation de chaque variable qui satisfait toutes les contraintes. Dans ce papier, sans manque de généralité, nous considérerons uniquement des contraintes binaires (i.e. des contraintes qui portent sur deux variables). Ainsi, la structure d'un CSP sera alors représentée par le graphe (X, C), appelé *graphe de contraintes*.

Le concept de base qui nous intéresse ici est la notion d'acyclicité des réseaux. En fait, et de façon surprenante, ce concept est généralement exprimé par le biais de la décomposition arborescente de graphes. Cependant, la décomposition arborescente de graphes s'avère restrictive, contrairement à celle de recouvrement de graphe par un hypergraphe acyclique. En effet, étant donné un graphe de contraintes, il peut exister plusieurs décompositions arborescentes qui correspondent à un même ensemble de clusters structurés en arbre, alors que pour ce même ensemble de clusters il n'existera qu'un unique hypergraphe acyclique. Par exemple, si l'on considère le graphe biparti complet $K_{1,3}$, cet (hyper)graphe possède plusieurs décompositions arborescentes. Le choix de l'une d'elles, en vue de la résolution, est alors *a priori* complètement arbitraire.

Pour rappel, pour un hypergraphe $H = (X, E)$, X est un ensemble fini de sommets, et $E = \{E_1, E_2, \dots E_m\}$ un ensemble d'arêtes (ou hyperarêtes) qui sont des sous-ensembles non vides de X. Ici,

nous ne considérerons que des hypergraphes réduits, c'est-à-dire des hypergraphes tels que pour toutes les arêtes E_i de H, E_i n'est pas un sous-ensemble d'une autre arête de H. Différentes définitions de l'acyclicité ont été proposées dans la littérature. Ici, nous nous référons à la définition classique de l'acyclicité d'hypergraphes, historiquement introduite sous le nom d'$\alpha - acyclicity$ dans [1] (qui correspond à la notion *d'hyper-arbre* dans [5]).

Définition 1 *Soit $G = (X, C)$ un graphe. Un recouvrement par un hypergraphe acyclique (CAH) du graphe G est un hypergraphe acyclique $H = (X, E)$ tel que pour chaque $\{x, y\} \in C$, il existe $E_i \in E$ tel que $\{x, y\} \subset E_i$. La largeur w d'un CAH (X, E) est égale à $max_{E_i \in E}|E_i| - 1$. La CAH-largeur w^* de G est la largeur minimale parmi tous les CAHs de G. Enfin, $\mathcal{CAH}(G)$ notera l'ensemble des CAHs de G.*

La notion de recouvrement par hypergraphe acyclique (appelé "hypertree embedding" dans [5]) est très proche de celle de décomposition arborescente qu'il est utile de rappeler ici. Une *décomposition arborescente* de $G = (X, C)$ est un couple (E, T) où $T = (I, F)$ est un arbre de nœuds I et d'arêtes F et $E = \{E_i : i \in I\}$ est une famille de sous-ensembles de X, telle que chaque sous-ensemble (appelé cluster) E_i est associé à un nœud de T et vérifie : (i) $\cup_{i \in I} E_i = X$, (ii) pour chaque arête $\{x, y\} \in C$, il existe $i \in I$ avec $\{x, y\} \subset E_i$, et (iii) pour tous $i, j, k \in I$, si k est dans une chaîne allant de i jusqu'à j dans T, alors $E_i \cap E_j \subset E_k$.

La largeur d'une décomposition arborescente (E, T) est égale à $max_{i \in I}|E_i| - 1$. La largeur arborescente (nous dirons *tree-width*) généralement notée w de G est la largeur minimale pour toutes les décompositions arborescentes de G.

Ainsi, il est facile de voir que pour une décomposition arborescente (E, T) de $G = (X, C)$, le couple (X, E) est un CAH du graphe G. De plus, la CAH-largeur w^* de G est égale à la *tree-width* de G. Cependant, le concept de CAH s'avère moins restrictif. En effet, pour un graphe donné et un CAH de ce dernier, il peut exister plusieurs décompositions arborescentes définies sur les hyperarêtes de ce CAH.

Etant donné un CSP doté d'un CAH de largeur w, la complexité en temps des meilleures méthodes structurelles, pour sa résolution, est $O(exp(w+1))$ alors que la complexité en espace peut être réduite à $O(exp(s))$ où s est la taille de la plus grande intersection $E_i \cap E_j$ (qui correspond à un séparateur) entre arêtes voisines de l'hypergraphe. Dans la suite, étant donné un graphe $G = (X, C)$ et l'un de ses CAH $H = (X, E)$, nous étudierons plusieurs classes de recouvrements acycliques de H. Ces recouvrements correspondent aux recouvrements d'hyperarêtes (éléments de E) par d'autres hyperarêtes (plus grandes mais *a priori* moins nombreuses), et qui appartiennent à un hypergraphe défini sur le même ensemble de sommets et qui est acyclique. Dans tous les cas, ces extensions seront définies par rapport à un CAH H particulier, appelé *CAH de référence*. Quand le problème considéré est exprimé lui-même par un hypergraphe, nous considérerons alors son graphe primal G (sa 2-section dans la terminologie de C. Berge).

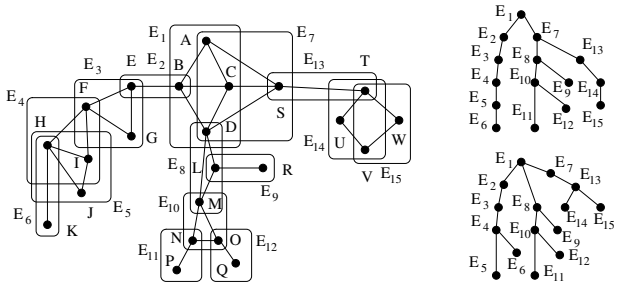

FIG. 1 – Un graphe de 23 sommets sous un recouvrement par un hypergraphe acyclique (de 15 arêtes) et 2 décompositions arborescentes possibles.

Définition 2 *L'ensemble des recouvrements d'un CAH $H = (X, E)$ d'un graphe $G = (X, C)$ est défini par $\mathcal{CAH}_{G,H} = \{(X, E') \in \mathcal{CAH}(G) : \forall E_i \in E, \exists E'_j \in E' : E_i \subset E'_j\}$.*

Les classes de recouvrements suivantes seront des restrictions successives de cette première classe $\mathcal{CAH}_{G,H}$. Mais avant de définir ces classes, nous allons définir les notions d'hyperarêtes voisines et de chaîne dans un hypergraphe acyclique $H = (X, E)$.

Définition 3 *Soient E_u et E_v deux hyperarêtes de H. E_u et E_v sont voisines si : $\nexists E_{i_1}, E_{i_2}, \ldots, E_{i_R}$ tel que $R > 2, E_{i_1} = E_u, E_{i_R} = E_v$ et $E_u \cap E_v \subsetneq E_{i_j} \cap E_{i_{j+1}}$, pour $j = 1, \ldots, R-1$.*
Une chaîne dans $H = (X, E)$ est une séquence d'arêtes $(E_{i_1}, \ldots E_{i_R})$ telle que $\forall j, 1 \leq j < R, E_{i_j}$ et $E_{i_{j+1}}$ sont voisines.

Un cycle dans $H = (X, E)$ est une chaîne $(E_{i_1}, E_{i_2}, \ldots E_{i_R})$ telle que $R > 3$ et $E_{i_1} = E_{i_R}$. La première restriction impose que les arêtes E_i recouvertes (éventuellement partiellement) par une même arête E'_j soient connexes dans H, i.e. mutuellement accessibles par des chaînes (dans la mesure où il n'est pas fait allusion à une orientation, nous utiliserons le terme chaîne plutôt que chemin). Cette classe est appelée *ensemble des recouvrements-connexes d'un CAH* et sera notée $\mathcal{CAH}_{G,H}[C^+]$. Il est possible de restreindre cette classe en conditionnant la nature de l'ensemble $\{E_{i_1}, E_{i_2}, \ldots E_{i_R}\}$.

D'une part, on peut limiter l'ensemble des chaînes considérées (classe des *recouvrements-chaînes d'un CAH* notée $\mathcal{CAH}_{G,H}[P^+]$), et d'autre part en prenant en compte la longueur maximum de connexion (classe des *recouvrements-famille d'un CAH* notée $\mathcal{CAH}_{G,H}[F^+]$). On peut aussi définir une classe (appelée *recouvrements-unique d'un CAH* et notée $\mathcal{CAH}_{G,H}[U^+]$) qui impose le recouvrement d'une arête E_i par une unique arête de E'. Finalement, il est possible d'étendre la classe $\mathcal{CAH}_{G,H}$ dans une autre direction (classe des *recouvrements-proches d'un CAH* notée $\mathcal{CAH}_{G,H}[B^+]$), en n'assurant ni la connexité, ni l'unicité : on peut recouvrir des arêtes dont l'intersection est vide mais qui possèdent une voisine commune.

Définition 4 *Etant donné un graphe G et un CAH H de G :*

- $\mathcal{CAH}_{G,H}[C^+] = \{(X,E') \in \mathcal{CAH}_{G,H} : \forall E'_i \in E', E'_i \subset E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_R}$ *avec* $E_{i_j} \in E$ *et* $\forall E_{i_u}, E_{i_v}, 1 \leq u < v \leq R$, *il y a une chaîne reliant* E_{i_u} *et* E_{i_v} *dans H définie sur les arêtes appartenant à* $\{E_{i_1}, E_{i_2}, \ldots E_{i_R}\}\}$.
- $\mathcal{CAH}_{G,H}[P^+] = \{(X,E') \in \mathcal{CAH}_{G,H} : \forall E'_i \in E', E'_i \subset E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_R}$ *avec* $E_{i_j} \in E$ *et* $(E_{i_1}, E_{i_2}, \ldots E_{i_R})$ *est une chaîne dans H*$\}$.
- $\mathcal{CAH}_{G,H}[F^+] = \{(X,E') \in \mathcal{CAH}_{G,H} : \forall E'_i \in E', E'_i \subset E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_R}$ *avec* $E_{i_j} \in E$ *et* $\exists E_{i_u} \in E, \forall E_{i_v}, 1 \leq v \leq R, E_{i_u}$ *et* E_{i_v} *sont voisines* $\}$.
- $\mathcal{CAH}_{G,H}[U^+] = \{(X,E') \in \mathcal{CAH}_{G,H} : \forall E_i \in E, \exists! E'_j \in E' : E_i \subset E'_j\}$.
- $\mathcal{CAH}_{G,H}[B^+] = \{(X,E') \in \mathcal{CAH}_{G,H} : \forall E'_i \in E', E'_i \subset E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_R}$ *avec* $E_{i_j} \in E$ *et* $\forall E_{i_u} \neq E_{i_v}, \exists E_{i_w} \in E \backslash \{E_{i_u}, E_{i_v}\}$ *: E_{i_u} et E_{i_w} sont voisines, E_{i_v} et E_{i_w} sont voisines* $\}$.

Si $\forall E'_i \in E', E'_i = E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_R}$, ces classes seront notées $\mathcal{CAH}_{G,H}[X]$ pour $X = C, P, F, U$ ou B.

Le concept de séparateur est essentiel dans les méthodes exploitant la structure, parce que la complexité en espace dépend directement de leur taille. Ce concept est introduit ici de sorte à imposer une nouvelle restriction. Cela rendra possible la limitation des séparateurs à un sous-ensemble de ceux qui existent dans l'hypergraphe de référence :

Définition 5 *L'ensemble des recouvrements-basés-séparateur d'un CAH $H = (X,E)$ est défini par* $\mathcal{CAH}_{G,H}[S] = \{(X,E') \in \mathcal{CAH}_{G,H} : \forall E'_i, E'_j \in E', i \neq j, \exists E_k, E_l \in E, k \neq l : E'_i \cap E'_j = E_k \cap E_l\}$.

En fait, cette classe impose l'unicité et la connexité. Elle correspond ainsi aux restrictions des classes $[U]$ et $[C]$. On peut ainsi déduire que l'unicité d'un recouvrement et la connexité des arêtes recouvertes imposent la conservation des séparateurs de H :

Théorème 1 $\mathcal{CAH}_{G,H}[S] = \mathcal{CAH}_{G,H}[U] \cap \mathcal{CAH}_{G,H}[C]$

Avant de donner la preuve de ce théorème, nous allons énoncer le lemme suivant.

Lemme 1 *Soient $H' \in \mathcal{CAH}_{G,H}[S]$, E_u, E_v deux hyperarêtes de H telles que $\exists E'_{u'}, E'_{v'}$, $E_u \subset E'_{u'}$ et $E_v \subset E'_{v'}$. Si E_u et E_v sont voisines alors $E'_{u'}$ et $E'_{v'}$ sont également voisines.*

Preuve On va supposer que $E'_{u'}$ et $E'_{v'}$ ne sont pas voisines : il existe donc une chaîne $(E'_{j_1}, E'_{j_2}, \ldots, E'_{j_R})$ telle que $E'_{j_1} = E'_{u'}$, $E'_{j_R} = E'_{v'}$ et $E'_{u'} \cap E'_{v'} \subsetneq E'_{j_l} \cap E'_{j_{l+1}}$, pour $l = 1, \ldots, R-1$. Puisque $E_u \subset E'_{u'}, E_v \subset E'_{v'}$ alors $E_u \cap E_v \subset E'_{u'} \cap E'_{v'}$. Etant donné que $H \in \mathcal{CAH}_{G,H}[S]$, pour $j = 1, \ldots, R-1, \exists E_{i_l}, E_{i_{l+1}}$ telles que $E_{i_l} \cap E_{i_{l+1}} = E'_{j_l} \cap E'_{j_{l+1}}$. Donc pour $l = 1, \ldots, R-1, E_u \cap E_v \subsetneq E_{i_l} \cap E_{i_{l+1}}$. Soit E_{i_l} et $E_{i_{l+1}}$ sont voisines, soit il existe une chaîne dans H qui les relie parce que H est connexe. On peut donc construire une chaîne qui part de E_{i_1} à E_{i_R} et dans laquelle les hyperarêtes de E_{i_l} à $E_{i_{l+1}}$ forment une chaîne. Le début et la fin de cette chaîne étant constitués d'hyperarêtes voisines, il existe donc un cycle dans H. Or H est acyclique donc notre hypothèse ($E'_{u'}$ et $E'_{v'}$ ne sont pas voisines) est fausse. □

Nous allons à présent donner une preuve du théorème 1.

Preuve On va montrer dans un premier temps que $\mathcal{CAH}_{G,H}[S] \subset \mathcal{CAH}_{G,H}[U] \cap \mathcal{CAH}_{G,H}[C]$.

Soit H' un hypergraphe recouvrant de $\mathcal{CAH}_{G,H}[S]$. On suppose que $H' \notin \mathcal{CAH}_{G,H}[U]$ alors $\exists E_i \in E, \exists E'_{j_1} \in E', \exists E'_{j_2} \in E' : E'_{j_1} \neq E'_{j_2}, E_i \subset E'_{j_1}$ et $E_i \subset E'_{j_2}$. Donc $E_i \subset E'_{j_1} \cap E'_{j_2}$. Or, E_i n'est pas une intersection entre hyperarêtes de H et donc $H' \notin \mathcal{CAH}_{G,H}[S]$. Ceci étant impossible, on a $H' \in \mathcal{CAH}_{G,H}[U]$. De même on suppose que $H' \notin \mathcal{CAH}_{G,H}[C] : \exists E'_i \in E', E'_i \subset E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_R}$ avec $E_{i_j} \in E$ et $\exists E_{i_u}, E_{i_v}, 1 \leq u < v \leq R$, il n'y a pas de chaîne dans H définie sur les arêtes appartenant à $\{E_{i_1}, E_{i_2}, \ldots E_{i_R}\}$ reliant E_{i_u} et E_{i_v}. Les hypergraphes dans $\mathcal{CAH}_{G,H}$ étant connexes, il existe dans H une chaîne $(E_{t_1}, \ldots, E_{t_Q})$ qui relie E_{i_u} et E_{i_v}, avec $E_{t_1} = E_{i_u}$ et $E_{t_Q} = E_{i_v}$. Cette chaîne contient au moins une hyperarête $E_{t_z} \notin \{E_{i_1}, E_{i_2}, \ldots E_{i_R}\}$. Donc la chaîne est recouverte dans H' par $E'_{t'_1}, \ldots, E'_{t'_{Q'}}$. On suppose que $E'_{t'_1} = E'_i$ et donc contient E_u et E_v. On pose $E'_{t'_k}$ une hyperarête contenant $E_{t_{p_{k-1}}}$ un élément de la chaîne $(E_{t_1}, \ldots, E_{t_Q})$ qui n'est pas inclus dans $E'_{t'_1}, \ldots, E'_{t'_{k-1}}$. On va montrer par récurrence sur k qu'il existe un élément $E_{t_{p_k}}$ de la chaîne qui n'est pas inclus dans $E'_{t'_1}, \ldots, E'_{t'_k}$. Soit $E_{t_{p_1}}$ le premier élément de la chaîne $(E_{t_1}, \ldots, E_{t_Q})$ qui n'est pas contenu dans

$E'_{t'_1}$ (existe parce qu'il y a au moins un élément de la chaîne non inclus dans E'_i). Donc il existe une hyperarête $E'_{t'_2}$ qui contient $E_{t_{p_1}}$: $E'_{t'_1}$ et $E'_{t'_2}$ sont voisines (lemme 1). Soit $E_{t_{p_2}}$ le premier élément de la chaîne $(E_{t_1}, \ldots, E_{t_Q})$ qui n'est pas contenu dans $E'_{t'_2}$ (le dernier E_v est dans $E'_{t'_1}$). Si $E_{t_{p_2}}$ est contenu dans $E'_{t'_1}$ alors $E'_{t'_1} \cap E'_{t'_1} = (E_{t_{p_1}} \cap E_{t_{p_1-1}}) \cup (E_{t_{p_2}} \cap E_{t_{p_2-1}})$ qui n'est pas une intersection entre hyperarêtes de H. Or $H' \in \mathcal{CAH}_{G,H}[S]$ donc $E_{t_{p_2}}$ n'est pas contenu dans $E'_{t'_1}$. Il existe alors $E'_{t'_3}$ contenant $E_{t_{p_2}}$: $E'_{t'_3}$ et $E'_{t'_2}$ sont voisines (lemme 1). Soit $E_{t_{p_3}}$ le premier élément de la chaîne $(E_{t_1}, \ldots, E_{t_Q})$ qui n'est pas contenu dans $E'_{t'_3}$. Si $E_{t_{p_3}}$ est contenu dans $E'_{t'_1}$ alors $E'_{t'_3}$ et $E'_{t'_1}$ sont voisines (lemme 1) : H' contient un cycle. Or ce dernier est acyclique, donc $E_{t_{p_3}}$ n'est pas contenu dans $E'_{t'_1}$. Si $E_{t_{p_3}}$ est contenu dans $E'_{t'_2}$ alors $E'_{t'_2} \cap E'_{t'_3} = (E_{t_{p_2}} \cap E_{t_{p_2-1}}) \cup (E_{t_{p_3}} \cap E_{t_{p_3-1}})$ qui n'est pas une intersection entre hyperarêtes de H. Or $H' \in \mathcal{CAH}_{G,H}[S]$ donc $E_{t_{p_3}}$ n'est pas contenu dans $E'_{t'_2}$. Donc il existe $E'_{t'_4}$ qui contient $E_{t_{p_3}}$. Pour $k > 3$, on va supposer que notre hypothèse est vérifiée pour tout $k' < k$ et on va le prouver pour k. Soit $E_{t_{p_k}}$ le premier élément de la chaîne $(E_{t_1}, \ldots, E_{t_Q})$ qui n'est pas contenu dans $E'_{t'_k}$. Si $E_{t_{p_k}}$ est contenu dans $E'_{t'_1}$ alors H' contient un cycle. Or ce dernier est acyclique, donc $E_{t_{p_k}}$ n'est pas contenu dans $E'_{t'_1}$. Si $E_{t_{p_k}}$ est contenu dans $E'_{t'_{k-1}}$ alors $E'_{t'_{k-1}} \cap E'_{t'_k} = (E_{t_{p_{k-1}-1}} \cap E_{t_{p_{k-1}}}) \cup (E_{t_{p_k-1}} \cap E_{t_{p_k}})$ qui n'est pas une intersection entre hyperarêtes de H. Or $H' \in \mathcal{CAH}_{G,H}[S]$ donc $E_{t_{p_k}}$ n'est pas contenu dans $E'_{t'_2}$. Si $E_{t_{p_k}}$ est contenu dans $E'_{t'_l}$, avec $1 < l < k-1$, alors H' contient un cycle. Or H' est acyclique, donc $E_{t_{p_k}}$ n'est pas contenu dans $E'_{t'_l}$. On a donc montré qu'il existe un élément de le chaîne qui n'est pas inclus dans $E'_{t'_l}$, avec $1 \leq l \leq k$. On vient de montrer l'existence d'une infinité d'hyperarêtes recouvrant un ensemble fini d'hyperarêtes formant la chaîne qui relie E_u et E_v. Ceci étant impossible, on a donc $H' \in \mathcal{CAH}_{G,H}[C]$.

Il nous reste à montrer que $\mathcal{CAH}_{G,H}[U] \cap \mathcal{CAH}_{G,H}[C] \subset \mathcal{CAH}_{G,H}[S]$. Soient un hypergraphe $H' \in \mathcal{CAH}_{G,H}[U] \cap \mathcal{CAH}_{G,H}[C]$, E'_i et E'_j deux hyperarêtes de H'. $E'_i = E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_R}$, $\{E_{i_1}, E_{i_2}, \ldots, E_{i_R}\}$ étant un ensemble connexe tel que chaque élément est uniquement recouvert par E'_i. De même $E'_j = E_{j_1} \cup E_{j_2} \cup \ldots \cup E_{j_Q}$, $\{E_{j_1}, E_{j_2}, \ldots, E_{j_Q}\}$ étant un ensemble connexe tel que chaque élément est uniquement recouvert par E'_j. Les deux ensembles sont disjoints. L'intersection entre E'_i et E'_j est constituée des intersections entre les éléments des deux ensembles. Si toutes ces intersections sont vides : $E'_i \cap E'_j = \emptyset$. Sinon il existe $E_{i_m} \in \{E_{i_1}, E_{i_2}, \ldots, E_{i_R}\}$ et $E_{j_n} \in \{E_{j_1}, E_{j_2}, \ldots, E_{j_Q}\}$ tels que $E_{i_m} \cap E_{j_n} \neq \emptyset$. S'il existe $E_{i_{m'}} \in \{E_{i_1}, E_{i_2}, \ldots, E_{i_R}\}$ et $E_{j_{n'}} \in \{E_{j_1}, E_{j_2}, \ldots, E_{j_Q}\}$ tels que $E_{i_{m'}} \cap E_{j_{n'}} \neq \emptyset$. On construit dans H une chaîne formée par la chaîne entre E_{i_m} et E_{j_n}, celle reliant E_{j_n} et $E_{j_{n'}}$ dans l'ensemble connexe $\{E_{j_1}, E_{j_2}, \ldots, E_{j_Q}\}$, $E_{j_{n'}}$, $E_{i_{m'}}$, la chaîne reliant $E_{i_{m'}}$ et E_{i_m} dans l'ensemble connexe $\{E_{i_1}, E_{i_2}, \ldots, E_{i_R}\}$. Soit cette chaîne forme un cycle ou bien l'une des deux intersections $E_{i_m} \cap E_{j_n}$, $E_{i_{m'}} \cap E_{j_{n'}}$ est incluse dans l'autre. H étant acyclique, alors une intersection contient l'autre : $E'_i \cap E'_j = E_{i_m} \cap E_{j_n}$ ou $E'_i \cap E'_j = E_{i_{m'}} \cap E_{j_{n'}}$. Donc $H' \in \mathcal{CAH}_{G,H}[S]$. □

Notons que cette classification n'est pas exhaustive. Nous pourrions considérer d'autres classes, mais dont l'intérêt (et par voie de conséquence l'étude) serait probablement limité. Notons pour finir que le calcul d'un élément de ces différentes classes est facile en termes de complexité. Par exemple, étant donnés G et H (H peut être obtenu à partir d'une décomposition arborescente), nous pouvons calculer $H' \in \mathcal{CAH}_{G,H}[S]$ en fusionnant des arêtes voisines dans H.

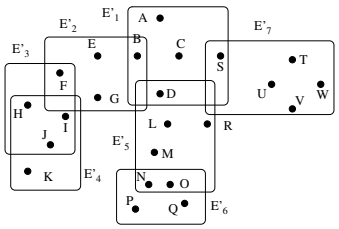

FIG. 2 – Dans ce recouvrement, ni la connexité ($E'_6 = E_{11} \cup E_{12}$), ni l'unicité ($E_5 \subset E'_4 \cap E'_3$) et donc les séparateurs (e.g. $E'_4 \cap E'_3$) ne sont vérifiés. Ce recouvrement appartient en fait à $\mathcal{CAH}_{G,H}[C^+]$.

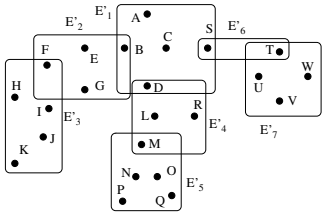

FIG. 3 – Ce recouvrement appartient à $\mathcal{CAH}_{G,H}[S]$.

Dans tous les cas, il est facile de constater que la valeur de la largeur des hypergraphes s'accroît à l'intérieur de ces classes, et qu'elle est donc plus grande que w, la largeur associée à l'hypergraphe de référence H. En particulier, pour les classes du type $\mathcal{CAH}_{G,H}[X^+]$, cet accroissement est assimilable à un accroissement additif :

Théorème 2 $\forall H' \in \mathcal{CAH}_{G,H}[X^+]$ *avec* $X = C, P, F, U$ *ou* B, $\exists \Delta \geq 0$ *tel que* $w' \leq w + \Delta$.

Concernant les autres classes, l'accroissement est multiplicatif. En effet, dans chaque cas, le recouvrement est relatif à la fusion d'arêtes de (X, E) :

Théorème 3 $\forall H' \in \mathcal{CAH}_{G,H}[C], \exists \delta \geq 1$ *tel que* $w' \leq \delta(w + 1 - s^-) + s^-$, *où* s^- *est la taille minimum des séparateurs.*

Pour les classes $\mathcal{CAH}_{G,H}[U]$ et $\mathcal{CAH}_{G,H}[B]$, les arêtes peuvent être déconnectées et par conséquent avec des intersections vides. Dans ce cas, nous ne pourrons pas prendre en compte la taille des séparateurs :

Théorème 4 $\forall H' \in \mathcal{CAH}_{G,H}[U] \cup \mathcal{CAH}_{G,H}[B], \exists \delta \geq 1$ *tel que* $w' \leq \delta(w + 1)$.

Ces remarques sont utiles parce qu'elles ont des conséquences sur la complexité des algorithmes qui exploiteront ces recouvrements. Elles illustrent en particulier le fait que les classes $\mathcal{CAH}_{G,H}[C]$ (et donc $[P]$, $[F]$ et $[S]$) doivent être privilégiées, au détriment de classes telles que $\mathcal{CAH}_{G,H}[U]$ ou $\mathcal{CAH}_{G,H}[B]$. Concernant la taille des séparateurs, on peut observer que pour la classe $\mathcal{CAH}_{G,H}[S]$, la valeur s associée à H constitue une borne supérieure pour tout hypergraphe H' considéré. Formellement :

Théorème 5 $\forall H' \in \mathcal{CAH}_{G,H}[S], s' \leq s$.

Cette étude nous indique les classes les plus prometteuses. D'un point de vue théorique, il semble que la classe $\mathcal{CAH}_{G,H}[S]$ devrait être la plus utile, à condition de limiter la taille des CAH-largeurs induites.

Dans la suite, nous allons exploiter ces concepts au niveau algorithmique. Aussi, chaque CAH sera maintenant doté d'une arête privilégiée - la racine - à partir de laquelle les différentes résolutions devront démarrer. Aussi, les connexions entre arêtes de l'hypergraphe seront orientées. Ainsi, certains concepts introduits plus haut seront maintenant exprimés par des mots tels que "hyperarête père", "hyperarête fils" ou "hyperarête frère" comme c'est le cas dans les arborescences.

3 Exploitation Algorithmique des CAHs

Plusieurs méthodes de résolution ont déjà été proposées pour exploiter les propriétés relatives à l'acyclicité de réseaux de contraintes. Ici, nous portons notre intérêt sur l'extension de BTD [10]. Cette méthode a d'une part démontré son intérêt pratique, et d'autre part, recèle l'avantage d'être facile à étendre à d'autres formalismes [3, 16, 19].

Cette méthode a été introduite en s'appuyant fortement sur la notion de décomposition arborescente de graphe. Nous allons constater que son adaptation aux hypergraphes acycliques est simple. BTD explore l'espace de recherche en affectant les variables relativement à un ordre induit par une décomposition arborescente orientée (E, T). BTD détermine d'abord un nœud E_1 racine de T, ce qui induit immédiatement une orientation. Pour un nœud E_i, $Père(E_i)$ désignera son nœud père et $Fils(E_i)$ l'ensemble de ses fils. Le sous-problème enraciné en E_i est le sous-problème induit par les variables des clusters contenus dans le sous-arbre enraciné en E_i. BTD réalise une recherche de type backtracking, en exploitant un ordre des variables induit par leur affectation (notée $\mathcal{A}$). Pendant la résolution, BTD affectera les variables V_{E_i} de E_i. Supposons que l'extension de l'affectation courante sur ces variables soit cohérente. Si E_i est une feuille de T, la résolution se poursuit sur une autre partie du problème. Si E_i possède des fils, BTD continuera la résolution sur l'un de ses fils $E_j \in Fils(E_i)$.

Notons que l'affectation $\mathcal{A}[E_i \cap E_j]$ permet de déconnecter le problème en deux sous-problèmes indépendants. L'un de ces sous-problèmes est localisé sous E_i et possède pour racine E_j. Il y a alors deux possibilités. Soit $\mathcal{A}[E_i \cap E_j]$ n'a jamais été calculée auparavant. Dans ce premier cas, la résolution se poursuit sur le sous-problème enraciné en E_j. Soit $\mathcal{A}[E_i \cap E_j]$ a déjà été calculée. Dans ce cas, si le sous-problème enraciné en E_j possède une extension consistante de $\mathcal{A}[E_i \cap E_j]$, cette information aura alors été enregistrée comme *good structurel de* E_i *par rapport à* E_j (i.e. une affectation consistante du séparateur $E_i \cap E_j$ qui peut être étendue de façon consistante à la totalité du sous-problème enraciné en E_j). Sinon, le sous-problème enraciné en E_j ne possède pas d'extension consistante de $\mathcal{A}[E_i \cap E_j]$, cette information aura alors été enregistrée comme *nogood structurel de* E_i *par rapport à* E_j (i.e. une affectation consistante du séparateur $E_i \cap E_j$ qui ne peut être étendue de façon consistante à la totalité du sous-problème enraciné en E_j). Dans chacun de ces deux cas, la résolution pourra immédiatement être stoppée sur cette partie du problème, soit par un succès (cas du good), soit par un échec (cas du nogood). L'efficacité de BTD est basée sur ces principes. La complexité en temps de BTD est $O(exp(w+1))$. Sa complexité en espace peut être bornée par $O(exp(s))$, étant entendu que cette borne ne sera jamais observée en pratique.

Nous proposons ici une extension de BTD, appelée *BDH* pour "Backtracking sur des recouvrements Dynamiques par Hypergraphes acycliques") et qui est donc basée sur une exploitation dynamique des CAH. Cette approche va rendre possible l'intégration d'heu-

ristiques d'ordonnancement des variables qui seront plus dynamiques. De telles heuristiques sont nécessaires pour s'assurer une résolution pratique efficace. Afin de faciliter l'implémentation et de garantir des bornes de complexité intéressantes, tant pour le temps que pour l'espace, nous considèrerons uniquement des hypergraphes recouvrants appartenant à $\mathcal{CAH}_{G,H}[S]$, pour lesquels G est le graphe de contraintes et H l'hypergraphe de référence.

En premier lieu, il nous faut considérer une orientation dans l'hypergraphe en distinguant une arête E_1 comme racine. les hyperarêtes voisines de E_1 sont ses fils et récursivement les hyperarêtes voisines de E_i sont ses fils à l'exception de celle qui se trouve déjà sur le chemin de la racine à E_i qui est en fait son père. Pour un nœud E_i, $Père(E_i)$ désignera son nœud père et $Fils(E_i)$ l'ensemble de ses fils. La descendance de E_i est l'ensemble des variables des hyperarêtes contenues dans le sous-hypergraphe acyclique enraciné en E_i. On appelle sous-problème enraciné en E_i le sous-problème induit par les variables de la descendance de E_i.

BDH possède 4 entrées : $\mathcal{A}$ l'affectation courante, E'_i l'arête courante, $V_{E'_i}$ l'ensemble des variables non affectées de E'_i, H' l'hypergraphe courant. Cet hypergraphe est calculé récursivement. Le choix (heuristique) d'un nouvel hypergraphe recouvrant H'' dans $\mathcal{CAH}_{G,H}[S]$ est réalisé avant le démarrage de la résolution sur un sous-problème. BDH résout récursivement les sous-problèmes enracinés dans E'_i et fournit en résultat *true* si $\mathcal{A}$ peut être étendue de façon consistante sur ce sous-problème et $false$ sinon. Lors de l'appel initial, l'affectation $\mathcal{A}$ est vide, le sous-problème enraciné en E_1 correspond à l'intégralité du problème et H est l'hypergraphe de référence. De manière analogue à BTD, l'ordre dans lequel les variables sont affectées est partiellement induit par l'hypergraphe courant H'. Pendant la résolution, l'hypergraphe recouvrant est modifié pour prendre en compte les évolutions du problème traité. Tant que $V_{E'_i}$ n'est pas vide, BDH choisit une variable x dans $V_{E'_i}$ (ligne 16) et une valeur dans son domaine (ligne 18) (s'il n'est pas vide) qui satisfait toutes les contraintes (en considérant aussi celles induites par les éventuels nogoods). $BDH(\mathcal{A} \cup \{x \leftarrow v\}, E'_i, V_{E'_i}\backslash\{x\}, H')$ est appelé sur le reste de l'hyperarête (ligne 19). Quand toutes les variables dans E'_i sont affectées, l'algorithme choisit un fils E'_j de l'hyperarête courante (ligne 4) (s'il en existe un). Si $\mathcal{A}[E'_i \cap E'_j]$ est un good (ligne 5), nous savons que $\mathcal{A}$ peut alors être étendue de façon consistante sur $Desc(E'_j)$ (descendance de E'_j). De même, si E'_i contient un séparateur $s_k = E_{k_u} \cap E_{k_{u'}}$ de H, avec $E_{k_{u'}}$ un fils de E_{k_u}, tel que le sous-problème enraciné en E'_j est inclus dans celui enraciné en $E_{k_{u'}}$ et si $\mathcal{A}[s_k]$ est un good (ligne 7), nous savons que $\mathcal{A}$ peut être étendue de façon consistante sur $Desc(E_{k_{u'}})$. Ainsi, $\mathcal{A}$ peut également être étendue de façon consistante sur $Desc(E'_j)$. Par conséquent, un forward-jump sera réalisé et l'algorithme poursuivra la résolution sur le reste du problème. Puisque les nogoods sont exploités comme des contraintes, l'affectation $\mathcal{A}$ sera stoppée dans E'_i si elle contient un nogood. Si E'_i ne contient pas de (no)good, alors la résolution se poursuit sur le sous-problème enraciné en E'_j. Si $\mathcal{A}$ admet une extension consistante sur ce sous-problème, $\mathcal{A}[E'_i \cap E'_j]$ est enregistrée comme good (ligne 11) et *true* est renvoyé comme résultat, sinon $\mathcal{A}[E'_i \cap E'_j]$ est enregistré comme nogood (ligne 12) et $false$ est renvoyé. Si BDH échoue lors de l'extension de $\mathcal{A}$ sur E'_i alors, il renvoie $false$.

Cette extension de BTD recèle plusieurs avantages. En premier lieu, elle permet de proposer de nombreuses heuristiques puisque l'on est libéré de la structure initiale H. Par exemple, la future variable à affecter n'est pas nécessairement choisie dans une unique hyperarête E_j. De plus, il est également possible d'exploiter tous les nogoods déjà trouvés, de même que de nombreux goods. La complexité en espace n'est pas modifiée ($O(exp(s))$) parce que la résolution s'appuie sur le même ensemble de séparateurs que ceux figurant dans H puisque nous considérons uniquement des hypergraphes recouvrants appartenant à $\mathcal{CAH}_{G,H}[S]$. Enfin, l'implémentation est facilitée toujours du fait que les hypergraphes considérés sont dans $\mathcal{CAH}_{G,H}[S]$ et que ces hypergraphes seront obtenus par une simple fusion d'arêtes voisines dans H.

Algorithme 1 : BDH($\mathcal{A}, E'_i, V_{E'_i}, H'$)

```
1  if V_{E'_i} = ∅ then
2      Cons ← true ; F ← Sons(E'_i)
3      while F ≠ ∅ and Cons do
4          Choose E'_j in F ; F ← F\{E'_j}
5          if A[E'_j ∩ E'_i] is a good then Cons ← true
6          else
7              if ∃ s_k = E_{k_u} ∩ E_{k_u'} in H s.t.
               E_{k_u'} ∈ Sons(E_{k_u}), s_k ⊂ E'_i, E'_j ⊂ Desc(E_{k_u'})
               and A[s_k] is a good then Cons ← true
8              else
9                  Choose H'' induced by H' and E'_j with root
                   E''_1
10                 Cons ← BDH(A, E''_1, E''_1\(E''_1 ∩ E'_i), H'')
11                 if Cons then Record the good A[E''_1 ∩ E'_i]
12                 else Record the nogood A[E''_1 ∩ E'_i]
13     if Cons then ∀E_u ∩ E_v ⊂ E'_i, record the good
       A[E_u ∩ E_v]
14     return Cons
15 else
16     Choose x ∈ V_{E'_i} ; d_x ← D_x ; Cons ← false
17     while d_x ≠ ∅ and ¬Cons do
18         Choose v in d_x ; d_x ← d_x\{v}
19         if A ∪ {x ← v} satisfies constraints and nogoods
           then Cons ← BDH(A ∪ {x ← v}, E'_i, V_{E'_i}\{x}, H')
20     return Cons
```

Théorème 6 *BDH est correct, complet et termine.*

BDH utilise un sous-ensemble d'hypergraphes de $\mathcal{CAH}_{G,H}[S]$. En utilisant le théorème 2, on sait qu'il existe $\Delta \geq 0$ tel que pour tout H' dans ce sous-ensemble, $w' \leq w + \Delta$. La complexité de la méthode dépend évidemment de Δ. En outre, ce facteur est paramétrable : il est possible de fixer la valeur de Δ et considérer uniquement les hypergraphes recouvrants de $\mathcal{CAH}_{G,H}[S]$ dont la largeur est bornée par $w + \Delta$. Dans tous les cas, la complexité de BDH est donnée par le théorème suivant.

Théorème 7 *La complexité en temps de BDH est* $O(exp(w + \Delta + 2))$.

Preuve Soient (X, D, C) un CSP, H le CAH de référence de $G = (X, C)$. Soit V un ensemble de $w+\Delta+2$ variables tel que $\exists E_{u_1}, \ldots, E_{u_r}$, une branche dans H (avec $r \geq 2$ puisque $|V| = w+\Delta+2$ et $w+1$ est la taille maximale des hyperarêtes de H), $V \subset E_{u_1} \cup \ldots \cup E_{u_r}$ et $E_{u_2} \cup \ldots \cup E_{u_{r-1}} \subsetneq V$ (resp. $E_{u_1} \cap E_{u_2} \subset V$) si $r \geq 3$ (resp. $r = 2$). Dans un premier temps, on va prouver que toute affectation sur V est générée une seule fois. $\forall H' \in \mathcal{CAH}_{G,H}[S], \exists E'_{i_1}, \ldots, E'_{i_{r'}}$, une branche, (avec $r' \geq 2$ puisque la taille maximale des hyperarêtes dans H' est $w + \Delta + 1$) telle que $V \subset E'_{i_1} \cup \ldots \cup E'_{i_{r'}}$ et $E'_{i_2} \cup \ldots \cup E'_{i_{r'-1}} \subsetneq V$ (resp. $E'_{i_1} \cap E'_{i_2} \subsetneq V$) si $r' \geq 3$ (resp. $r' = 2$). Soit $\mathcal{A}$ une affectation à étendre sur V. L'ordre dans lequel les variables de $\mathcal{A}$ seront instanciées est induit par $H' \in \mathcal{CAH}_{G,H}[S]$. On supposera dans la suite que les hyperarêtes recouvrant V sont ordonnées suivant leur ordre d'affectation : E'_{i_j} est affectée avant $E'_{i_{j'}}$ si $j < j'$. De ce fait E_{u_1} est la première arête affectée parmi celles sur la branche de H recouvrant V et $s_1 = E_{u_1} \cap E_{u_2}$ est incluse dans E'_{i_1} parce que $E_{u_1} \subset E'_{i_1}$. Si $E'_{i_1} \cap E'_{i_2} = s_1$, la résolution du sous-problème enraciné en E'_{i_2} avec l'affectation $\mathcal{A}$ conduit à l'enregistrement d'un (no)good sur le séparateur s_1 : $\mathcal{A}[s_1]$. Soit $\mathcal{B}$ une nouvelle affectation à étendre sur V avec les mêmes valeurs de $\mathcal{A}[V]$ et $H'' \in \mathcal{CAH}_{G,H}[S]$ induit l'ordre dans lequel les variables de $\mathcal{B}$ sont affectées. $\exists E''_{j_1}, \ldots, E''_{j_{r''}}, r'' \geq 2$, une branche de H'' telle que $V \subset E''_{j_1} \cup \ldots \cup E''_{j_{r''}}$ et $E''_{j_2} \cup \ldots \cup E''_{j_{r''-1}} \subset V$ (resp. $E''_{j_1} \cap E''_{j_2} \subset V$) si $r'' \geq 3$ (resp. $r' = 2$). Etant donné que $s_1 \subset E''_{j_1}$, dès que E''_{j_1} est totalement affectée, $\mathcal{A}[s_1]$ arrête l'affectation des variables de V. Alors $\mathcal{A}[V]$ est générée une deuxième fois sur uniquement $w + \Delta + 1$ variables de V au pire.

On suppose maintenant que $E'_{i_1} \cap E'_{i_2} \neq s_1$. Si $\mathcal{A}[E'_{i_1} \cap E'_{i_2}]$ peut être étendue de manière consistante sur le sous-problème enraciné en E'_{i_2} alors $\mathcal{A}[E'_{i_1} \cap E'_{i_2}]$ est enregistrée comme un good. Puisque $s_1 \subset E'_{i_1}$, si l'affectation courante $\mathcal{A}$ peut être étendue de manière consistante sur les variables non instanciées du sous-problème enraciné en E'_{i_1} alors $\mathcal{A}[s_1]$ est enregistrée comme un good. Sinon, cette affectation ne peut être étendue de manière consistante sur au moins un sous-problème enraciné en $E'_{t'}$ une hyperarête de H' telle que $E'_{t'} \neq E'_{i_l}, l = 1, \ldots, r'$ et $E'_{t'} \cap E'_{i_1} \neq \emptyset$. Donc $\mathcal{A}[E'_{i_1} \cap E'_{t'}]$ est enregistrée comme un nogood. Si on essaie d'étendre $\mathcal{B}$ sur V, si $\mathcal{A}[s_1]$ est un good alors pour les mêmes raisons qu'auparavant $\mathcal{A}[V]$ est générée une deuxième fois sur uniquement $w + \Delta + 1$ variables de V au pire.

Si $\mathcal{A}[s_1]$ n'est pas un good alors $\mathcal{A}[E'_{i_1} \cap E'_{t'}]$ est enregistrée comme un nogood. Néanmoins, il est possible de générer $\mathcal{A}[V]$ une deuxième fois si $((E'_{i_1} \cap E'_{i_2}) \cup (E'_{i_1} \cap E'_{t'})) \subset E''_{j_{r''}}$. Si $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ peut être étendue de manière consistante sur le sous-problème enraciné en E''_{j_2} alors $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ est enregistrée comme un good. Puisque $s_1 \subset E''_{j_1}$, soit $\mathcal{B}[s_1]$ est enregistrée comme un good soit $\mathcal{A}[E''_{j_1} \cap E''_{t''}]$ est enregistrée comme un nogood, avec $E''_{t''}$ une hyperarête de H'' telle que $E''_{t''} \neq E''_{j_l}, l = 1, \ldots, r''$ et $E''_{t''} \cap E''_{j_1} \neq \emptyset$. Si $\mathcal{B}[s_1]$ est un good alors $\mathcal{B}[V]$ est générée à nouveau sur uniquement $w + \Delta + 1$ variables de V au pire.

Si $\mathcal{B}[E''_{j_1} \cap E''_{t''}]$ est un nogood : deux nogoods sont enregistrés sur deux séparateurs dans V. Dès que le premier de ces deux séparateurs est totalement instancié, le nogood associé à ce séparateur arrête l'affectation sur V. De ce fait $\mathcal{A}[V]$ n'est pas générée à nouveau.

Si $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ ne peut être étendue de manière consistante sur le sous-problème enraciné en E''_{j_2} alors $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ est enregistré comme un nogood. Puisque deux nogoods sont enregistrés, $\mathcal{A}[V]$ n'est pas générée à nouveau.

Si $\mathcal{A}[E'_{i_1} \cap E'_{i_2}]$ ne peut être étendue de manière consistante sur le sous-problème enraciné en E'_{i_2} alors $\mathcal{A}[E'_{i_1} \cap E'_{i_2}]$ est enregistrée comme un nogood. Si on essaie d'étendre $\mathcal{B}$ sur V, il est possible de générer $\mathcal{A}[V]$ une deuxième fois si $E'_{i_1} \cap E'_{i_2} \subsetneq E''_{j_{r''}}$. Si $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ peut être étendue de manière consistante sur le sous-problème enraciné en E''_{j_2} alors $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ est enregistrée comme un good. Soit $\mathcal{B}[s_1]$ est enregistrée comme un good, soit $\mathcal{B}[E''_{j_1} \cap E''_{t''}]$ est enregistrée comme un nogood. Dans tous les cas, $\mathcal{B}[V] = \mathcal{A}[V]$ n'est pas générée une nouvelle fois.

Si $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ ne peut être étendue de manière consistante sur le sous-problème enraciné en E''_{j_2} alors $\mathcal{B}[E''_{j_1} \cap E''_{j_2}]$ est enregistrée comme un nogood. Deux nogoods étant enregistrés, $\mathcal{A}[V]$ n'est pas générée une nouvelle fois.

On a prouvé donc que toute affectation sur V est générée au pire deux fois. Maintenant, on considère que H est recouvert par un ensemble de V_i vérifiant V_i contient $w + \Delta + 2$ variables telles que $\exists E_{u_1}, \ldots, E_{u_r}$, une branche dans H (avec $r \geq 2$), $V_i \subset E_{u_1} \cup \ldots \cup E_{u_r}$ et $E_{u_2} \cup \ldots \cup E_{u_{r-1}} \subsetneq V_i$ (resp. $E_{u_1} \cap E_{u_2} \subsetneq V_i$) si $r \geq 3$

(resp. $r = 2$). Il suffit de recouvrir chaque branche de H par des V_i (ils peuvent s'intersecter). Sur chacun des V_i recouvrant H une affectation est produite au plus deux fois. Le nombre d'affectations possibles sur V_i est $d^{w+\Delta+2}$. Donc le nombre d'affectations possibles sur l'ensemble des variables du problème est borné par $nombre_{V_i}.d^{w+\Delta+2}$, avec $nombre_{V_i}$ le nombre d'ensembles V_i recouvrant H. Le nombre de V_i étant borné, la complexité de BDH est en $O(exp(w + \Delta + 2))$. □

4 Etude Expérimentale

Dans cette section, nous évaluons l'efficacité pratique de BDH sur les benchmarks présentés dans [9]. Ces benchmarks sont des classes de CSP structurés avec un certain nombre de paramètres (n, d, w, t, s, ns, p) dont les instances sont générées suivant le modèle donné dans [9]. Une instance est définie par un ensemble n de variables qui ont des domaines de taille d. Son graphe de contraintes est un arbre de cliques avec ns sommets de taille au plus $w + 1$ et des tailles de séparateurs bornées par s. t donne le nombre de couples de valeurs interdits pour chaque contrainte. p est le pourcentage d'arêtes retirées de l'instance ainsi construite. Les expérimentations sont menées sur un PC sur linux avec un processeur Pentium 4 3,2 GHz et 1 Go de mémoire. Pour chaque classe d'instances, les résultats présentés sont les moyennes sur un ensemble de 50 instances. Le temps de résolution est limité à 30 minutes par instance. Au-delà le solveur est arrêté et le problème correspondant est considéré comme non résolu. Pour ces problèmes non résolus, on considère dans le calcul de la moyenne que le temps de résolution est de 30 minutes.

L'hypergraphe de référence est calculé grâce à une triangulation du graphe de contraintes G par la méthode MCS [18]. Nous reprenons également les meilleures heuristiques données dans [9] pour ce qui concerne le parcours de l'hypergraphe. Elles sont toutes basées sur la notion d'espérance mathématique du nombre de solutions [17]. On utilise pour le choix de racine l'heuristique $minesp$ (l'hyperarête racine est celle qui minimise le ratio entre l'espérance du nombre de solutions et la taille des hyperarêtes) pour l'ordonnancement des hyperarêtes fils $minesp_f$ (les fils sont ordonnés suivant la valeur croissante du ratio entre l'espérance du nombre de solutions et la taille des hyperarêtes).
Pour exploiter efficacement BDH, nous devons maintenant définir des heuristiques pour calculer dynamiquement des hypergraphes recouvrants. Afin de réduire la complexité en espace, nous fusionnerons les hyperarêtes avec les heuristiques suivantes, uniquement si la taille de leur intersection est supérieure à 5 et si la taille des hyperarêtes résultantes est inférieure à $1,5w$ (Δ est borné par $0,5w$). En respectant l'ordre dans lequel les hyperarêtes sont considérées, différents hypergraphes recouvrants sont calculés. Br : l'hyperarête courante est fusionnée avec son premier fils (par rapport à $minesp_f$), le premier fils étant celui dont le score est le plus petit et ainsi de suite ..., si la taille de leurs séparateurs est supérieure à 5 et tant que la taille des hyperarêtes résultantes est inférieure à $1,5w$. $Br+vp$: tout d'abord, l'hypergraphe de référence est modifié de sorte que les hyperarêtes avec moins de 3 variables non incluses dans leur parent soient fusionnées avec. Ensuite, l'heuristique Br est utilisée pour calculer dynamiquement les hypergraphes recouvrants. La table 1 montre les temps d'exécution de BDH avec les heuristiques Br, $Br + vp$, avec les heuristiques basées sur $minesp$ dans les Classes 3 et 4 définies dans [9] (avec un hypergraphe dont les intersections entre hyperarêtes, de taille maximum est bornée par 5 pour la Classe 4). La Classe 4 obtient de meilleurs résultats que la Classe 3. L'heuristique Br a de très bons résultats, proches de ceux de la Classe 4, et permet de résoudre toutes les instances de $(250, 20, 20, 99, 10, 25, 10)$ alors que deux de ces instances ne sont pas résolues dans la Classe 4 en moins de 1 800 s.

Disposer d'un meilleur hypergraphe de référence, au sens de la résolution, permet à l'heuristique $Br + vp$ d'obtenir généralement les meilleurs résultats. Ces expérimentations mettent en valeur l'importance considérable que recèle la connaissance d'un hypergraphe de référence de qualité pour la résolution. Enfin, une exploitation dynamique de cette structure d'hypergraphe conduit à des améliorations significatives.

Il est important de noter que lorsque BDH utilise des heuristiques de la Classe 3, il est équivalent à BTD. De ce fait, il obtient des résultats similaires à ceux obtenus par BTD dans [9] pour cette classe. En outre, les méthodes FC et MAC ne parviennent pas à résoudre la plupart de ces instances en moins de 1 800 s.

5 Conclusion

Dans ce papier, nous avons proposé un cadre basé sur un vieux concept : le recouvrement de problème par des hypergraphes acycliques. Nous avons montré qu'une telle approche semble être plus utile que celles fondées sur la décomposition arborescente.

Dans un premier temps, ces recouvrements ont été étudiés d'un point de vue théorique, afin de déterminer leurs caractéristiques et propriétés. Après, nous nous sommes focalisés sur une classe de recouvrements particulière, qui conserve les résultats de complexité en temps et en espace, et qui permet aussi d'en améliorer d'autres. A notre sens, l'un des résultats essentiels

CSP (n, d, w, t, s, ns, p)	(a)	(b)	(c)	(d)
$(150, 25, 15, 215, 5, 15, 10)$	2,04	1,78	1,93	1,76
$(150, 25, 15, 237, 5, 15, 20)$	8,84	1,23	1,12	1,10
$(150, 25, 15, 257, 5, 15, 30)$	4,20	1,40	1,30	3,19
$(150, 25, 15, 285, 5, 15, 40)$	3,08	1,17	0,27	0,23
$(250, 20, 20, 107, 5, 20, 10)$	14,87	10,49	12,89	13,14
$(250, 20, 20, 117, 5, 20, 20)$	11,43	6,97	13,18	5,98
$(250, 20, 20, 129, 5, 20, 30)$	29,24	3,62	2,80	2,87
$(250, 20, 20, 146, 5, 20, 40)$	12,26	6,99	3,97	7,89
$(250, 25, 15, 211, 5, 25, 10)$	7,18	4,19	3,79	4,64
$(250, 25, 15, 230, 5, 25, 20)$	3,86	2,76	1,69	2,36
$(250, 25, 15, 253, 5, 25, 30)$	3,86	3,73	2,61	2,08
$(250, 25, 15, 280, 5, 25, 40)$	56,66	13,87	11,88	12,33
$(250, 20, 20, 99, 10, 25, 10)$	112,88	82,07	78,16	144,20
$(500, 20, 15, 123, 5, 50, 10)$	7,11	2,23	2,00	3,12
$(500, 20, 15, 136, 5, 50, 20)$	6,11	3,01	3,23	4,13

TAB. 1 – Temps d'exécution (en s) sur des CSP aléatoires partiellement structurés : (a) Classe 3, (b) Classe Br, (c) Classe $Br + vp$ et (d) Classe 4.

présentés dans ce papier porte sur le théorème 7 qui indique que l'ajout de Δ variables pour offrir plus de liberté dans le choix dynamique de l'ordre des variables, induit une complexité en temps limité, précisément $O(exp(w + \Delta + 2))$. Cette borne améliore significativement les résultats connus. De plus, nous avons montré que notre approche facilite l'implémentation et permet en outre d'améliorer les résultats pratiques.

Enfin, pour faciliter sa présentation, ce travail est exposé dans le cadre des CSPs. Néanmoins, notre approche est applicable à d'autres formalismes tels que SAT et au-delà MAX-SAT, MAX-CSP, VCSP et les réseaux probabilistes.

Remerciements

Ce travail a été soutenu par un programme blanc de l'ANR (projet STAL-DEC-OPT).

Références

[1] C. Beeri, R. Fagin, D. Maier, and M. Yannakakis. On the desirability of acyclic database schemes. *J. ACM*, 30 :479–513, 1983.

[2] A. Darwiche. Recursive conditioning. *Artificial Intelligence*, 126 :5–41, 2001.

[3] S. de Givry, T. Schiex, and G. Verfaillie. Exploiting Tree Decomposition and Soft Local Consistency in Weighted CSP. In *Proc. of AAAI*, pages 22–27, 2006.

[4] R. Dechter. Bucket Elimination : A Unifying Framework for Reasoning. *Artificial Intelligence*, 113(1-2) :41–85, 1999.

[5] R. Dechter. *Constraint processing*. Morgan Kaufmann Publishers, 2003.

[6] R. Dechter and J. Pearl. Tree-Clustering for Constraint Networks. *Artificial Intelligence*, 38 :353–366, 1989.

[7] G. Gottlob, N. Leone, and F. Scarcello. Hypertree Decompositions and Tractable Queries. *J. Comput. Syst. Sci.*, 64(3) :579–627, 2002.

[8] J. Huang and A. Darwiche. A structure-based variable ordering heuristic for SAT. In *Proc. of IJCAI*, pages 1167–1172, 2003.

[9] P. Jégou, S.N. Ndiaye, and C. Terrioux. 'Dynamic Heuristics for Backtrack Search on Tree-Decomposition of CSPs. In *Proc. of IJCAI*, pages 112–117, 2007.

[10] P. Jégou and C. Terrioux. Hybrid backtracking bounded by tree-decomposition of constraint networks. *Artificial Intelligence*, 146 :43–75, 2003.

[11] P. Jégou and C. Terrioux. Decomposition and good recording for solving Max-CSPs. In *Proc. of ECAI*, pages 196–200, 2004.

[12] W. Li and P. van Beek. Guiding Real-World SAT Solving with Dynamic Hypergraph Separator Decomposition. In *Proc. of ICTAI*, pages 542–548, 2004.

[13] R. Marinescu and R. Dechter. Dynamic Orderings for AND/OR Branch-and-Bound Search in Graphical Models. In *Proc. of ECAI*, pages 138–142, 2006.

[14] I. Rish and R. Dechter. Resolution versus Search : Two Strategies for SAT. *Journal of Automated Reasoning*, 24 :225–275, 2000.

[15] N. Robertson and P.D. Seymour. Graph minors II : Algorithmic aspects of tree-width. *Algorithms*, 7 :309–322, 1986.

[16] M. Sachenbacher and B. C. Williams. Bounded Search and Symbolic Inference for Constraint Optimization. In *Proc. of IJCAI*, pages 286–291, 2005.

[17] B. Smith. The Phase Transition and the Mushy Region in Constraint Satisfaction Problems. In *Proceedings of ECAI*, pages 100–104, 1994.

[18] R. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13 (3) :566–579, 1984.

[19] C. Terrioux and P. Jégou. Bounded backtracking for the valued constraint satisfaction problems. In *Proc. of CP*, pages 709–723, 2003.