Les CSP extrêmaux

Nicolas Prcovic

Les CSP extrêmaux

Nicolas Prcovic
LSIS - Université Paul Cézanne - Aix-Marseille III
nicolas.prcovic@lsis.org

Résumé

Nous proposons une nouvelle classe de CSP binaires appelés CSP extrêmaux. Les CSP de cette classe sont inconsistants mais deviendraient consistants si n'importe quel couple de valeurs interdit devenait autorisé. Etant inconsistants, ils ne sont pas traitables avec des méthodes de réparation locale. Comme ils autorisent un nombre très élevé de solutions partielles presque complètes, ils peuvent être très difficiles à résoudre à l'aide de méthodes de recherche arborescente intégrant le filtrage des domaines. Il faudra donc trouver de nouvelles méthodes pour les résoudre. Nous présentons un algorithme simple de génération de CSP extrêmaux. Nous constatons expérimentalement que les CSP extrêmaux équilibrés sont beaucoup plus longs à résoudre que les CSP aléatoires dits difficiles de même taille. Nous présentons aussi un schéma d'algorithme susceptible d'être performant sur des problèmes difficiles, dès lors qu'on sera en mesure de générer suffisamment rapidement des CSP extrêmaux.

Abstract

We present a new class of binary CSPs called extremal CSPs. The CSPs of this class are inconsistent but would become consistent if any pair of variable assignments among the forbidden ones was allowed. Being inconsistent, they cannot be solved by any local repair method. As they allow a great number of partial (almost complete) solutions, they can be very hard to solve with tree search methods integrating domain filtering. We design a simple algorithm for generating extremal CSPs. We experiment that they are much harder to solve than random CSPs of same size at the complexity peak. We show how the concept of extremality can be useful to define new means of deciding the consistency of CSPs. In particular, we propose a new algorithm which should be efficient on the hardest CSPs if we can find a way of generating extremal CSPs quickly.

1 Introduction

Bien que la résolution de problème de satisfaction de contraintes soit un problème NP-complet, on sait que seule une petite partie de l'ensemble des CSP est difficile à résoudre avec les méthodes utilisées jusqu'à présent. Lorsque le problème contient beaucoup de solutions assez uniformément réparties dans l'espace de recherche, les méthodes de réparation locale sont efficaces. Lorsqu'un problème est fortement surcontraint, une recherche arborescente associée à une méthode incomplète de détermination d'inconsistance de solutions partielles peut permettre de montrer rapidement que le problème n'a pas de solution. Ce sont les problèmes qui résistent à ces deux types de méthodes qui sont habituellement considérés comme difficiles.

Des caractérisations de problèmes difficiles ont été trouvés pour les CSP aléatoires définis par un certain nombre de paramètres (nombre de variables, taille des domaines, densité du graphe et dureté des contraintes). Dans ce modèle, les CSP dont le paramétrage fait qu'ils ont une probabilité de 0.5 de posséder une solution sont en moyenne plus longs à résoudre que les autres. L'inconvénient de cette caractérisation est qu'elle ne donne pas une propriété qui garantit la difficulté du problème mais seulement une probabilité plus forte qu'il soit difficile à résoudre. Par ailleurs, il a été proposé d'autres propriétés générales pouvant expliquer la difficulté de résolution de certains problèmes : taille du "backbone", de la "backdoor" minimale, du noyau inconsistant, etc. Mais ces propriétés ne permettent pas de décrire précisément la structure d'un problème difficile.

Ce que nous proposons ici, c'est une description simple mais très précise de ce que sont les CSP les plus difficiles à traiter avec les méthodes de résolution actuelles. Pour ceci, nous expliquons d'abord ce qui fait qu'un problème est intrinséquement difficile à résoudre grâce aux techniques actuelles (section 2), puis nous

introduisons la classe des CSP extrêmaux, qui inclut les problèmes les plus difficiles (section 3). Nous donnons des propriétés des CSP extrêmaux et indiquons en quoi les étudier peut aider à définir de nouvelles méthodes pour décider de la consistance de CSP. Ensuite, nous indiquons comment générer les CSP extrêmaux (section 4). Puis, nous générons des CSP extrêmaux équilibrés, les plus difficiles de leur classe, et vérifions expérimentalement qu'à taille de problème égale les CSP extrêmaux équilibrés sont beaucoup plus longs à résoudre que les CSP aléatoires au pic de complexité (section 5).

2 Les CSP difficiles

Nous nous plaçons dans le cadre des CSP binaires, chacun défini par un triplet (X, D, C), où X=$\{x_1$, ..., $x_n\}$ est l'ensemble des variables, D=$\{D_1$, ..., $D_n\}$ est l'ensemble des domaines discrets et finis de valeurs que peuvent prendre les variables et C est l'ensemble des contraintes, contenant tous les couples d'affectations de variables *autorisés*. Nous dirons que le CSP est consistant s'il possède une solution (un ensemble A d'affectations de toutes ces variables tels que tout couple d'affectations de A appartient à C).

Les procédures de recherche arborescente que nous considérons sont celles qui filtrent dynamiquement les domaines en maintenant un certain niveau de consistance partielle, tel que Forward-Checking, MAC, etc. Ces procédures peuvent inclure des améliorations qui utilisent le retour-arrière intelligent (Backjumping), l'apprentissage par l'échec (Nogood recording, Dynamic backtracking), etc.

Définition 1 *Arbre de recherche minimal (d'une procédure de recherche)*
Etant donnée une procédure de recherche arborescente, l'arbre de recherche minimal d'un CSP inconsistant est celui qui résulte d'une suite de choix (dynamiques) optimaux de prochaine variable à affecter qui fait que le nombre global de noeuds générés est minimal.

En d'autres termes, parmi l'ensemble des arbres de recherche que l'on peut obtenir en variant à un moment ou à un autre un certain nombre de choix de variable, l'arbre minimal est celui qui contient le moins de nœuds. Cet arbre de recherche est un arbre de preuve d'inconsistance, chaque nœud représentant une étape élémentaire de la preuve. Cette étape élémentaire doit être de complexité polynomiale et elle dépend de la procédure de résolution mise en oeuvre. Un problème est donc difficile s'il n'existe pas de preuve courte de son inconsistance.

Nous pourrions généraliser la définition de l'arbre de recherche minimal pour qu'elle englobe aussi les CSP consistants. Mais ça n'aurait aucun intérêt par rapport à notre définition de la difficulté car, pour tout problème consistant, il existe une preuve de longueur linéaire en fonction du nombre de variables qui montre sa consistance : celle qui consiste à effectuer une suite d'affectations de variables avec les valeurs d'une solution. En conséquence, nous considérerons que les CSP consistants sont intrinséquement faciles pour la raison qu'il existe une preuve courte de leur consistance[1].

Définition 2 *Difficulté des CSP inconsistants*
Un CSP inconsistant C1 est plus difficile qu'un CSP inconsistant C2 pour une procédure de recherche arborescente donnée si l'arbre de recherche minimal de C1 est plus grand que celui de C2.

Notre but est de mettre à jour les CSP qui sont difficiles pour toute procédure de recherche arborescente. Remarquons que les procédures qui intègrent le retour arrière intelligent ou l'apprentissage par l'échec permettent essentiellement de pallier les effets néfastes de mauvais choix de variables. Le Backjumping "backtracke" comme si la dernière variable avait été choisie juste après les autres variables coupables des échecs sur cette variable. Le Dynamic backtracking réordonne les variables a posteriori pour retrouver l'ordre qu'il aurait mieux valu choisir. En conséquence, meilleure est l'heuristique de choix de variable, plus le gain apporté par ces mécanismes est limité. C'est pourquoi nous simplifierons notre étude en ne considérant que les méthodes de recherche qui incluent seulement des techniques (prédictives) de filtrage des domaines.

3 CSP extrêmaux

Pour des ensembles de variables X et de domaines D donnés, nous allons présenter une classe de CSP qui a la propriété informelle suivante : tout CSP possédant aussi l'ensemble de variables X et l'ensemble de domaines D mais qui n'est pas dans cette classe est tel qu'il existe toujours un CSP de cette classe qui est plus difficile que lui, quel que soit le niveau de consistance partielle maintenu par la procédure de recherche arborescente utilisée. L'idée est qu'il suffit de savoir résoudre tous les CSP de cette classe pour être capable de résoudre n'importe quel CSP qui possède ces mêmes variables et ces mêmes domaines. Nous donnons maintenant une définition précise des membres de cette classe, appelés problèmes extrêmaux.

[1] Il est bien évident qu'en pratique il ne suffit pas qu'il existe une preuve courte pour qu'elle soit facile à trouver. La notion de difficulté que nous introduisons ici diffère de celle qui est habituellement utilisée, qui est liée au temps moyen de résolution et non à la longueur minimale de la preuve.

Définition 3 *CSP extrêmal*
Un CSP est dit extrêmal *s'il est inconsistant et si l'ajout de n'importe quel couple d'affectations de variable autorisé dans l'ensemble des contraintes le rendrait consistant.*

Les CSP extrêmaux sont saturés de solutions partielles impliquant $n-1$ affectations de variables. Ils sont donc potentiellement difficiles à résoudre par des méthodes de recherche arborescente car les méthodes de filtrages de domaines par maintien d'un certain niveau de cohérence (cohérence d'arc, de chemin, etc) peuvent souvent s'avérer totalement inopérantes. Leur difficulté relative s'explique ainsi :

- Si on ajoute un couple d'affectations de variable à un CSP extrêmal, il possède au moins une solution et donc une méthode de recherche arborescente ou de réparation locale est susceptible de trouver une solution rapidement.
- Si on lui retire un couple d'affectations de variable, on introduit des possibilités supplémentaires de backtracker (à cause de ce couple devenu interdit) à la méthode de recherche arborescente et on lui évite l'exploration de certains sous-arbres. En d'autres termes, on a ajouté un nogood (ie, une contrainte redondante) donc la preuve de l'inconsistance peut être raccourcie.

Pour ces raisons, chaque CSP extrêmal est normalement plus difficile à traiter que beaucoup d'autres CSP : ceux qui autorisent les mêmes couples de valeurs mais auxquels on a soit ajouté soit retiré des couples de valeurs autorisés. Bien évidemment, le fait qu'un CSP soit extrêmal ne garantit pas qu'il soit intrinséquement difficile mais il garantit une difficulté relative par rapport à tout un ensemble d'autres CSP de même taille.

Nous allons maintenant transposer le modèle de CSP sous la forme d'un problème sur les graphes afin de faciliter l'appréhension et la caractérisation des CSP extrêmaux. Les CSP binaires sont représentables sous forme de graphes simples colorés par ce qu'on appelle leur microstructure.

Définition 4 *Microstructure d'un CSP*
La microstructure d'un CSP est un graphe coloré dont chaque sommet correspond à une affectation d'une de ses variables avec une valeur de son domaine, dont chaque arête correspond à un couple de valeurs autorisé et dont chaque sommet est coloré par son numéro de variable.

La microstructure d'un CSP possédant n variables est un graphe n-partite n-coloré (chaque partie a une couleur spécifique). Il est consistant s'il contient une n-clique (un sous-graphe complet à n sommets). Nous pouvons donc maintenant redéfinir les CSP extrêmaux dans le cadre des graphes.

Définition 5 *CSP extrêmal (2)*
Un CSP est dit extrêmal *si sa microstructure est telle qu'elle ne contient aucune n-clique mais que l'ajout de n'importe quelle arête entre deux sommets de couleur différente ferait apparaître une n-clique.*

A titre d'exemple, nous donnons en figure 1 les 4 structures possibles de CSP extrêmaux lorsqu'il y a 3 variables dont chacun des domaines est de taille 2.

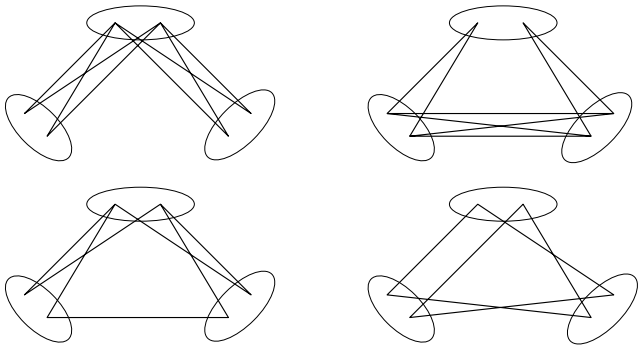

FIG. 1 – Les 4 structures de CSP extrêmaux ayant 3 variables dont les domaines sont de taille 2. Les ellipses entourent les sommets de même couleur. Il n'y a aucun triangle. Toute nouvelle arête liant deux sommets de couleur différente créerait au moins un triangle.

La *théorie des graphes extrêmaux* est l'étude de la façon dont la structure intrinsèque de graphes leur assure l'existence de certaines propriétés, telle que l'existence de cliques ou la coloriabilité. Par exemple, le théorème de Turan [16] montre l'unicité de T(n,k), le graphe à n sommets sans k-clique qui maximise le nombre d'arêtes. Ce qui signifie qu'un graphe à n sommets ayant plus d'arêtes contient au moins une k-clique. La proposition 2 donne un résultat similaire (mais il n'y a pas unicité du graphe) pour les graphes n-partite sans n-clique. Les microstructures de CSP extrêmaux sont ainsi une forme de graphes extrêmaux. A notre connaissance, leurs propriétés n'ont pas encore été étudiées.

3.1 Les CSP extrêmaux pour décider de la consistance

La considération des CSP extrêmaux permet d'obtenir un nouveau moyen de décider de la consistance d'un CSP.

Proposition 1 *Un CSP est inconsistant si et seulement si sa microstructure est un graphe partiel (au sens large) de la microstructure d'un CSP extrêmal ayant les mêmes variables et les mêmes domaines.*

Preuve 1 *Il est évident que tout CSP dont la microstructure est un graphe partiel de celle d'un CSP extrêmal est inconsistant car ce CSP extrêmal est lui-même inconsistant. Considérons maintenant un CSP inconsistant. Soit on ne peut ajouter une arête à sa microstructure de telle façon que la microstructure résultante soit celle d'un CSP inconsistant et alors on avait affaire à un CSP extrêmal, soit on peut ajouter une arête qui maintienne l'inconsistance et on obtient un nouveau graphe. Dans ce dernier cas, on peut réitérer la tentative d'ajout d'arête préservant l'inconsistance. Mais au bout d'un nombre fini d'itérations, on aboutira forcément à une microstructure de CSP consistant car toute microstructure est un graphe partiel du graphe n-partite complet (consistant). Donc toute microstructure de CSP inconsistant est un graphe partiel de microstructure de CSP extrêmal.*

En admettant qu'il est plus difficile de montrer qu'il n'existe pas de n-clique dans un graphe que dans un de ses graphes partiels, nous pouvons considérer que si nous disposons d'une méthode efficace sur les CSP extrêmaux alors elle doit être encore plus efficace sur tous les autres CSP.

3.2 Les CSP extrêmaux pour montrer la consistance

L'étude des propriétés des CSP extrêmaux peut permettre d'obtenir des moyens de déterminer rapidement la consistance de certains CSP. En voici un exemple simple.

Proposition 2 *Le nombre maximum d'arêtes dans la microstructure d'un CSP extrêmal ayant n variables et dont les domaines sont D_1, ..., D_n est $(\sum_{i=1}^{i=n-1} \sum_{j=i+1}^{j=n} |D_i|.|D_j|) - d_1.d_2$ où d_1 et d_2 sont les tailles des deux plus petits domaines.*

Preuve 2 *Considérons le CSP n'ayant aucune contrainte dont les domaines sont D_1, ..., D_n, c'est-à-dire celui dont la microstructure est un graphe n-partite complet. Ce graphe contient $\sum_{i=1}^{i=n-1} \sum_{j=i+1}^{j=n} |D_i|.|D_j|$ arêtes et $\prod_{i=1}^{i=n} |D_i|$ n-cliques (ie, solutions). Si on retire une arête entre deux sommets dont les tailles des parties sont $|D_i|$ et $|D_j|$, on fait disparaître $\frac{\prod_{k=1}^{k=n} |D_k|}{|D_i|.|D_j|}$ n-cliques. Pour maximiser le nombre de n-cliques enlevées, il faut minimiser le produit $|D_i|.|D_j|$, c'est-à-dire choisir les plus petits domaines, dont les tailles seront notées d_1 et d_2. Lorsqu'on va retirer d'autres arêtes, on ne pourra pas retirer plus de n-cliques que la première fois. Mais on pourra en retirer autant : il suffit de retirer une nouvelle arête liant des sommets associés aux mêmes variables que pour la première arête. Lorsqu'on aura retiré toutes les arêtes entre ces deux variables, il n'y aura plus de n-cliques et il restera $(\sum_{i=1}^{i=n-1} \sum_{j=i+1}^{j=n} |D_i|.|D_j|) - d_1.d_2$ arêtes dans la microstruture.*

La microstructure en haut à gauche de la figure 1 présente une telle microstructure de CSP extrêmal dont on a retiré toutes les arêtes entre deux variables à partir du graphe tripartite complet. Remarquons que plusieurs types de CSP extrêmaux peuvent avoir le nombre maximal d'arêtes. Par exemple, celui en haut à droite de la figure 1 possède aussi 8 arêtes.

Cette propriété portant sur les CSP extrêmaux nous permet d'en obtenir une sur la consistance des CSP généraux.

Proposition 3 *Tout CSP ayant n variables, dont les domaines sont D_1, ..., D_n et dont la microstructure contient strictement plus de $A = (\sum_{i=1}^{i=n-1} \sum_{j=i+1}^{j=n} |D_i|.|D_j|) - d_1.d_2$ arêtes, où d_1 et d_2 sont les tailles des deux plus petits domaines, est consistant.*

Preuve 3 *D'après la proposition 2, toute microstructure de CSP ayant plus de A arêtes n'est pas celle d'un CSP extrêmal et n'est pas non plus celle d'un de ses graphes partiels, donc, d'après la proposition 1, elle n'est pas la microstructure d'un CSP inconsistant.*

Nous avons donc une méthode de complexité temporelle en $\Theta(n^2)$ qui peut permettre d'établir qu'un CSP est consistant sans trouver de solution. Evidemment, cette méthode ne peut pas déterminer l'inconsistance de CSP et elle n'est capable d'établir la consistance que d'un nombre très restreint de CSP. Nous l'avons évoqué ici pour illustrer la possibilité d'avoir des méthodes de complexité faible permettant de déterminer la consistance de CSP en étudiant les CSP extrêmaux. Des propriétés de CSP extrêmaux plus utiles restent à trouver.

3.3 Les CSP extrêmaux pour montrer l'inconsistance

Maintenant, nous allons voir qu'on peut aussi définir des méthodes de preuve d'inconsistance en se basant sur les CSP extrêmaux. Lorsqu'on fait une recherche arborescente pour résoudre un CSP, on cherche à déterminer si sa microstructure contient une n-clique. Si on trouve une n-clique, on s'arrête car on a établi la consistance. Mais ce n'est qu'après avoir épuisé toutes les combinaisons de sommets que l'on peut établir l'inconsistance. Grâce à la notion de CSP extrêmal, nous pouvons définir un algorithme qui essaie de déterminer si la microstructure d'un CSP est un graphe partiel de microstructure d'un des CSP extrêmaux qui ont

les mêmes variables et domaines. Un tel algorithme peut s'arrêter et conclure à l'inconsistance du CSP dès qu'un CSP extrêmal dont il est le graphe partiel est trouvé. La figure 2 présente cet algorithme. Il présuppose l'existence d'une procédure d'énumération des CSP extrêmaux qui ont le même ensemble de domaines que le CSP à résoudre, ainsi qu'une procédure de test d'isomorphisme de graphe partiel. Cet algorithme considère chacun des CSP extrêmaux et s'arrête en concluant à l'inconsistance dès que le test d'isomorphisme de graphe partiel est positif. Par contre, si le CSP est consistant, il le vérifie en énumérant tous les CSP extrêmaux associés et en constatant qu'il n'est le graphe partiel d'aucun d'entre eux. L'intérêt de cet algorithme est que l'inconsistance peut être déterminée très tôt, dès les premiers tests d'isomorphisme de graphe partiel, si on est capable d'énumérer les CSP extrêmaux dans un ordre choisi par une bonne heuristique.

```
procedure est-inconsistant(P)
   pourtout G ∈ Frontiere(P) faire
      si P est isomorphe à un graphe partiel de G
         retourne VRAI
   retourne FAUX
```

FIG. 2 – La procédure `est-inconsitant(P)`, qui permet de déterminer l'inconsistance d'un CSP en générant des CSP extrêmaux.

Remarquons quand même que le problème du test d'isomorphisme de graphe partiel, qui est un cas particulier du test d'isomorphisme de sous-graphe, est un problème NP-complet. Cependant, il est d'autant plus facile que le graphe partiel est proche du graphe de référence. En particulier, le problème du test d'isomorphisme de graphe[2] est considéré comme plus facile (et est résolu très efficacement en pratique) bien que personne n'ait démontré jusqu'à présent s'il est NP-complet ou pas. Or, la possibilité de résoudre efficacement des CSP inconsistants qui sont proches des CSP extrêmaux nous intéressent particulièrement car ils font partie de ceux qui posent le plus de difficultés aux méthodes de recherche arborescente usuelles. Cet algorithme est donc susceptible d'être plus efficace sur les problèmes considérés habituellement comme difficiles et inefficaces sur les problèmes qu'on sait déjà facilement résoudre. Mais avant tout, pour appliquer cette méthode, il nous faudrait savoir énumérer rapidement les CSP extrêmaux. Ce problème est non trivial, comme nous allons le voir dans la section suivante.

[2]qui détermine si deux graphes sont isomorphes ou pas.

4 Génération des CSP extrêmaux

L'algorithme de test d'inconsistance de la figure 2 nécessite une procédure efficace de génération de CSP extrêmaux pour être elle-même efficace. D'une manière générale, la génération efficace de graphes ayant certaines propriétés est un problème qui peut être plus ou moins difficile en fonction de la propriété de ces graphes. La difficulté est de ne pas générer de graphes isomorphes entre eux car ils sont équivalents. Deux graphes sont isomorphes s'il existe une bijection des sommets de l'un vers les sommets de l'autre et que cette bijection appliquée à l'ensemble des arêtes de l'un donne les arêtes de l'autre. Une classe d'isomorphisme représente l'ensemble des graphes isomorphes entre eux. Une procédure de génération de graphes respectant certaines propriétés (dans notre contexte : être une microstructure de CSP extrêmal) ne doit idéalement générer qu'un seul représentant, appelé *graphe canonique*, par classe d'isomorphisme. C'est d'autant plus crucial que le cardinal d'une classe d'isomorphisme contenant des graphes de n sommets est $n!$ (le nombre de permutations de l'ensemble des sommets qui produisent un graphe différent) si ses graphes ne contiennent pas d'automorphisme (de permutation qui laisse le graphe inchangé), ce qui est le cas de la majorité des graphes. Les seules classes d'isomorphisme qui ne contiennent qu'un représentant sont celles contenant un graphe complet ou un graphe sans arête. Par exemple, la classe d'isomorphisme de CSP extrêmaux qui est représentée en bas à droite de la figure 1 contient le CSP dont tous les domaines sont {1, 2} et dont toutes les contraintes sont des contraintes de différences. Elle contient aussi les trois CSP avec les mêmes domaines mais une seule contrainte de différence et deux contraintes d'égalité. Pour passer du premier CSP à l'un des trois autres, il suffit de permuter deux valeurs du domaine de l'une des trois variables. Les 4 classes d'isomorphisme des CSP extrêmaux à 3 variables dont les domaines sont $\{1, 2\}$ contiennent en tout 33 CSP différents.

Il existe des procédures générales de génération de graphes sans isomorphe qui imposent des conditions sur la canonicité, les plus connus étant les *orderly algorithm* [14], mais ils n'explicitent pas un test de canonicité qui soit efficace. A notre connaissance, un tel test efficace n'a pas encore été trouvé dans un contexte général (si tant est qu'il existe). Par contre, il existe des procédures spécialisées pour générer efficacement les graphes qui ont certaines propriétés : les arbres, les graphes cubiques (ie, dont tous les sommets ont degré 3) [2] et plus généralement, les graphes ayant des propriétés héréditaires[3] [8]. Nous n'avons pas encore

[3]Une propriété d'un graphe est héréditaire si tous ses sous-

trouvé d'algorithme efficace de génération de CSP extrêmaux et cette partie du travail reste donc à faire.

Par contre, il est beaucoup plus facile de trouver un algorithme de génération de CSP extrêmaux si on ne se soucie ni de sa rapidité ni du problème de redondance dû aux isomorphismes. Un tel algorithme sera quand même utile pour la raison suivante. La classe des problèmes extrêmaux inclut l'ensemble des problèmes les plus difficiles à résoudre par les méthodes arborescentes habituelles. Si des améliorations de ces méthodes sont capables de résoudre efficacement tous les CSP extrêmaux alors on pourra considérer que c'est aussi vrai pour tous les CSP inconsistants. Or, nous avons tout notre temps pour générer des séries de CSP extrêmaux qui pourraient servir de bibliothèques de problèmes destinés à éprouver les techniques de résolution traditionnelles ou leurs améliorations.

```
procedure genere-extrêmaux(P, A)
    S = cherche-solution(P)
    si S = nil
        affiche P
        retourne VRAI
    sinon
        pourtout e ∈ S ∩ A faire
            si genere-extrêmaux(P\{e}, A\S) = VRAI
                retourne VRAI
        retourne FAUX
```

FIG. 3 – La procédure `genere-extrêmaux(P, A)`, qui permet de générer des CSP extrêmaux qui sont des graphes partiels de P.

La procédure `genere-extrêmaux(P, A)` (figure 3) permet de générer des CSP extrêmaux qui sont des graphes partiels de P. Au départ P et A sont tous les deux le même graphe n-partite complet. La procédure consiste à chercher itérativement une solution au problème et à la faire disparaître de ce problème, jusqu'à ce qu'il ne reste plus de solution. Si P contient une n-clique S, on essaie toutes les façons de retirer une arête de $S \cap A$, où A est l'ensemble des arêtes qu'on s'autorise encore à retirer. Puis, on retire une arête de P et on retire de A toutes les arêtes de S. De cette manière, on sait que si on rajoutait cette arête plus tard, on ferait à nouveau apparaître S puisque les autres arêtes qui forment S seront toujours présentes. Ainsi, après avoir retiré un certain nombre d'arêtes selon ce schéma, lorsqu'on arrive à une microstructure qui n'a plus de n-clique, on garantit que toute arête qu'on ajouterait à la microstructure ferait apparaître (au moins) une n-clique, ce qui correspond bien à un CSP extrêmal. Notons deux inconvénients à notre procédure : elle n'a pas de garantie de complétude (uniquement de correction) et elle est inutilisable par la procédure `est-inconsitant(P)` car elle utilise elle-même une méthode déjà capable de détecter l'inconsistance de CSP extrêmal. En pratique, elle génère tous les CSP extrêmaux ayant 3 variables de taille 2. Nous n'avons pu vérifier si elle restait complète avec des valeurs plus élevées à cause du nombre extrêmement grand de CSP extrêmaux qui en résulte. Quoi qu'il en soit, celà nous suffit pour générer autant de CSP extrêmaux que l'on veut pour créer des CSP difficiles à résoudre par les méthodes arborescentes existantes.

graphes possèdent aussi cette propriété.

5 Les CSP extrêmaux difficiles

Nous avons tout d'abord essayé de générer des CSP extrêmaux de telle manière que le retrait d'arête soit fait aléatoirement. Les temps de résolution obtenus étaient toujours très faibles. Ceci s'explique facilement. Même si la définition d'un CSP extrêmal garantit que si un CSP n'est pas extrêmal alors il existe un CSP extrêmal qui est plus difficile à résoudre que lui par une méthode de recherche arborescente, il existe des CSP extrêmaux qui sont plus faciles à résoudre que des CSP non extrêmaux. Par exemple, le CSP extrêmal qui correspond à un graphe n-partite dont on a retiré toutes les arêtes entre deux parties (cf le graphe en haut à gauche de la figure 1) est un CSP qui interdit tout couple de valeurs entre deux des variables. Une simple recherche en profondeur d'abord sans filtrage mais avec une heuristique de choix de variable qui choisit la variable de plus faible degré est capable de le résoudre en backtrackant systématiquement sur la deuxième variable choisie. Par ailleurs, seul le CSP représenté en bas à droite de la figure 1 est arc-consistant. Par contre, lorsque les CSP extrêmaux ont plus de variables et des domaines de plus grande taille, presque tous ces CSP ont un degré de cohérence partielle élevé : chaque valeur a de nombreux supports dans tous les domaines puisqu'elle appartient à de nombreuses (n-1)-cliques.

Comme nous l'avons vu, pour qu'un CSP (extrêmal ou non) soit le plus difficile possible, il faut que son temps de résolution soit le moins sensible possible à l'ordre de choix des variables effectués par une recherche arborscente. Sinon, il y aurait de gros écarts de temps de résolution et cela impliquerait qu'il existe un temps de résolution faible pour un ordre de choix de variable donné. Le problème ne serait donc pas difficile en soi. A l'opposé, s'il est complètement insensible à l'ordre du choix de variable, c'est qu'il est complétement symétrique (comme le CSP en bas à droite de la figure 1). En effet, tout choix entre des alternatives

symétriques est indifférent car il mène à des situations équivalentes. Afin de générer des CSP extrêmaux réellement difficiles, nous avons fait en sorte de minimiser l'écart de degré entre leurs sommets de plus haut et de plus bas degré. Cela s'intègre très facilement dans la procédure `genere-extrêmaux` en faisant en sorte de choisir de retirer de P en premier l'arête qui maintient l'écart de degré le plus petit. L'équilibre entre les degrés des sommets favorise l'apparition des symétries mais ne la garantit pas du tout.

Afin de vérifier la difficulté des CSP extrêmaux équilibrés, après les avoir généré grâce à notre procédure, nous les avons résolu à l'aide d'une recherche arborescente classique : un Forward-Checking avec l'heuristique de choix de variable dynamique dom/deg (taille du domaine divisé par le degré de la variable dans le graphe de contraintes). Le tableau 1 indique le nombre moyen de retours arrières et le nombre moyen de nœuds de l'arbre de recherche des 100 premiers CSP extrêmaux générés pour chaque paramétrage (nombre de variables, taille des domaines). En regard, nous avons indiqué les résultats (avec la même procédure de résolution) pour les CSP aléatoires de [7] de même taille les plus difficiles, c'est-à-dire ceux dont le graphe de contraintes est complet et dont la valeur de dureté des contraintes maximise le temps moyen de résolution (problèmes au pic de complexité). Les temps de résolution ne sont pas indiqués car ils sont presque toujours trop faibles pour être mesurés (inférieurs à 0.01 seconde) sauf pour les CSP extrêmaux avec 25 variables qui ont des temps moyens de résolution de 0.04 secondes.

Nous pouvons constater que les CSP extrêmaux équilibrés que nous générons sont beaucoup plus difficiles à résoudre que les CSP aléatoires générés au seuil, à nombre de variables et à taille de domaine égaux. Nous avons donc bien trouvé une méthode de génération de problèmes plus difficiles (c'est-à-dire, plus long à résoudre en moyenne) que les CSP aléatoires les plus difficiles générés de façon classique.

L'autre caractéristique des CSP que nous avons obtenu est que leur difficulté n'est pas forcément une fonction croissante du nombre de variables et de la taille de leurs domaines. Nous avons conjecturé la cause suivante. Comme l'égalité de degré entre les sommets est une condition de symétrie nécessaire mais non suffisante, notre procédure a pu obtenir des CSP très symétriques dans certains cas mais pas symétriques dans d'autres cas. Pour vérifier cette conjecture, nous avons déterminé le nombre d'orbites des microstructures générées, grâce à Nauty [9]. L'orbite d'un sommet dans un graphe est l'ensemble des sommets dont il peut être l'image par un automorphisme (une permutation des sommets qui laisse le graphe inchangé), c'est-à-dire l'ensemble des sommets interchangeables auquel ce sommet appartient. Ainsi, plus le nombre d'orbites dans un graphe est grand, moins il contient de symétries. Si le nombre d'orbites est égal au nombre de sommets, il n'y a pas de symétrie. Expérimentalement, nous constatons en effet que les CSP particulièrement difficiles sont ceux qui ont un certain nombre de symétries : les problèmes ayant 25 variables de domaine de taille 4 ont 90 orbites en moyenne (ce qui signifie que 10 de leurs sommets appartiennent à des symétries) tandis que les autres problèmes n'ont pas de symétries. La connaissance de ces symétries peut être extrêmement utile pour accélérer la résolution de ces CSP grâce à des méthodes de traitement de symétries [4, 12, 10, 1, 5, 13]. Cependant, il n'en reste pas moins que les CSP qui n'ont pas de symétries sont quand même bien plus difficiles que les CSP aléatoires de même taille. Sans avoir de symétrie, ils possèdent une certaine régularité et un niveau de consistance partielle très élevé qui met en échec les heuristiques de choix de variables et les techniques de filtrage de domaines.

6 Travaux connexes

La recherche de propriétés caractérisant la difficulté des CSP ou des problèmes SAT est très active depuis une dizaine d'années. Un certain nombre de travaux essaient de déterminer quels paramétrages de CSP aléatoires rendent les problèmes difficiles et pourquoi [17, 15]. D'autres travaux remarquent une grande variabilité dans le temps de résolution de certains problèmes consistants. Dans [6], on montre qu'il est possible d'éviter à une recherche de se perdre dans un grand sous-espace sans solution en introduisant de l'aléatoire dans certains choix de valeurs et en arrêtant puis en relançant la procédure plusieurs fois. Certains problèmes consistants qui paraissaient difficiles ne le sont donc plus du moment qu'on s'autorise à revenir sur certains choix de valeurs avant la fin de l'exploration d'un sous-arbre. Mais, dans [11] est définie la notion de "backbone" qui caractérise l'ensemble des affectations de variables que contiennent toutes les solutions d'un problème. Il y est montré que plus la taille du "backbone" est grande, plus le problème est difficile. Les problèmes consistants qui ont un grand "backbone" sont donc intrinsèquement plus difficiles que les autres car il y a un grand nombre de variables pour lesquelles une seule valeur correcte est possible si on veut obtenir une solution.

La notion de "backdoor" [18] est utilisée pour caractériser une affectation d'un sous-ensemble des variables du problème qui permet à une procédure de faible complexité de conclure à la consistance du pro-

		CSP extrêmaux équilibrés		CSP aléatoires difficiles	
n	d	# BT	# nœuds	# BT	# nœuds
13	8	251 (247-255)	503 (499-507)	11 (0-22)	93 (13-183)
17	6	115 (108-142)	365 (323-512)	10 (0-20)	99 (17-185)
22	5	78 (6-209)	223 (36-575)	7 (0-17)	94 (22-185)
25	4	12821 (5049-30070)	43022 (20948-102431)	6 (5-10)	50 (32-70)

TAB. 1 – Comparaisons du nombre moyen de retours-arrière (BT) et de nœuds pour différentes tailles de problèmes entre les 100 premiers CSP extrêmaux générés par la procédure `genere-extrêmaux` et 100 instances des CSP aléatoires les plus difficiles. (Les résultats moyens sont suivis de leur intervalle de valeurs entre parenthèses.

blème. Plus la plus "backdoor" de taille minimum est grande, plus le problème est difficile pour la méthode de résolution utilisée. D'autres travaux s'intéressent aux problèmes inconsistants, comme dans le présent article. Le concept le plus proche de celui de CSP extrêmal est peut-être celui de noyau insatisfiable (unsatisfiable core) pour SAT [3]. Un noyau insatisfiable minimum est le plus petit sous-ensemble de clauses insatisfiable d'un problème SAT. Plus le noyau insatisfiable minimum est grand, plus le nombre d'étapes pour résoudre le problème est élevé. Un problème SAT difficile peut donc être défini comme un problème dont toutes les clauses sont utiles à la démonstration de son insatisfiabilité, ce qui est le cas lorsque toutes ses clauses constituent le noyau insatisfiable minimum. Lui retirer une clause le rendrait satisfiable, tout comme ajouter un couple de valeurs autorisé rend un CSP extrêmal consistant.

Remarquons que toutes ces propriétés de problèmes difficiles ne permettent pas de caractériser leur structure aussi finement que pour les CSP extrêmaux.

7 Conclusion et perspectives

Dans le but de caractériser précisément ce qu'était un CSP difficile, nous avons introduit la classe des CSP extrêmaux. Nous avons montré expérimentalement qu'en générant des CSP extrêmaux dont les sommets de la microstructure ont des degrés voisins, on peut obtenir des problèmes beaucoup plus difficiles à résoudre que des CSP aléatoires de même taille au pic de complexité. Les CSP extrêmaux équilibrés sont conçus pour mettre en échec à peu près toutes les techniques de résolution proposées jusqu'à présent : réparation locale, recherche arborescente avec filtrage, retour arrière intelligent, heuristique de choix de variable ou de valeur, traitement des symétries. Ils constituent donc un défi et nécessitent l'élaboration de techniques de résolution nouvelles capables de les traiter efficacement. Dans cette perspective, nous avons donné une définition de la consistance d'un CSP en fonction de la possibilité que la microstructure d'un CSP soit un graphe partiel de CSP extrêmal. Cela nous a conduit à évoquer un nouveau schéma d'algorithme pour décider de la consistance de CSP. L'efficacité de ce schéma sur les CSP difficiles reste pour l'instant spéculative et nécessite qu'on sache générer rapidement des CSP extrêmaux. C'est cette question cruciale qu'il nous faudra maintenant traiter.

Références

[1] Rolf Backofen and Sebastian Will. Excluding symmetries in constraint-based search. In *Principles and Practice of Constraint Programming*, pages 73–87, 1999.

[2] G. Brinkmann. Fast generation of cubic graphs. *J. Graph Theory*, 23 :139–149, 1996.

[3] V. Chvtal and E. Szemeredi. Many hard examples for resolution. *Journal of the ACM*, 1988.

[4] Torsten Fahle, Stefan Shamberger, and Meinhof Sellman. Symmetry breaking. In *Proceedings of CP'01*, pages 93–107, 2001.

[5] Ian Gent and Barbara Smith. Symmetry breaking during search in constraint programming. In *proceedings of ECAI*, 2000.

[6] C. P. Gomes, B. Selman, N. Crato, and H. Kautz. Heavy-tail phenomena in satisfiability and constraint satisfaction. *Journal of Automated Reasoning*, 2000.

[7] T. Hulubei. *The CSP Library*. University of New Hampshire, 1999. http ://www.cs.unh.edu/ tudor/csp/.

[8] B. D. McKay. Isomorph-free exhaustive generation. *J. Algorithms*, 26(2) :306–324, 1998.

[9] Brendan D. McKay. Practical graph isomorphism. *Congressus Numerantium*, 30 :45–87, 1981.

[10] P. Meseguer and C. Torras. Exploiting symmetries within constraint satisfaction search. *Artificial Intelligence*, 29(1-2) :133–163, 2001.

[11] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. Determining computational complexity from characteristic 'phase transitions'. *Nature*, 1999.

[12] J. Pearson Pascal Van Hentenrick, P. Flener and M. Agren. Tractable symmetry breaking for csps with interchangeable values. In *proceedings of IJCAI 03*, pages 277–282, 2003.

[13] Jean François Puget. Symmetry breaking revisited. In *proceedings of CP'02*, 2000.

[14] Ronald C. Read. Every one a winner or how to avoid isomorphism search when cataloguing combinatorial configurations. *Annals of Discrete Mathematics*, 2 :107–120, 1978.

[15] B. M. Smith. Phase transition and the mushy region in constraint satisfaction problems. *11th European Conference on Artificial Intelligence*, pages 100–104, 1994.

[16] P. Turan. On an extremal problem in graph theory. *Mat. Fiz. Lapok*, 1941.

[17] C. P. Williams and T. Hogg. Using deep structure to locate hard problems. In *Proc. AAAI'92*, 1992.

[18] R. Williams, C. P. Gomes, and B. Selman. Backdoors to typical case complexity. In *Proc. of IJCAI'03*, 2003.