Cohérence d'arc pour contraintes molles

Thomas Schiex
INRA, Toulouse, France
`www-bia.inra.fr/T/schiex`

Résumé

Les cohérences locales sont définies comme des relaxations de la cohérence qui peuvent être vérifiées en temps polynomial. Elles sont accompagnées d'algorithmes de « filtrage » qui calculent, en temps polynomial, un CSP équivalent satisfaisant la propriété de cohérence locale.

L'éventuelle extension de telles propriétés à des cadres où les contraintes peuvent être violées (*soft constraints* ou contraintes molles) a ete abordée dans de nombreux papiers, dans de nombreux cas [BMR95, SFV95, RHZ76].Le principal résultat est que la notion de cohérence locale peut être étendue aux contraintes molles quand l'opérateur de combinaison des violations est idempotent. Dans le cas où l'opérateur n'est pas idempotent, cas des MAX-CSP, les choses sont moins claires et différentes notions de cohérence d'arc ont été proposées [SFV95, Wal94, AB98, LMSV98, LMS99]. Aucune d'entre elles ne satisfait vraiment les propriétés d'une cohérence locale.

Dans ce travail, nous montrons que via l'ajout d'un simple axiome, vérifié par la grande majorité des cadres existants (en particulier les MAX-CSP), il est possible de définir une notion de cohérence d'arc qui conserve les bonnes propriétés de la cohérence d'arc classique, à l'exception de l'unicité de la fermeture arc cohérente. Nous montrons que le filtrage par cette cohérence d'arc fournit des minorants plus forts que les minorants déjà proposés dans la littérature. Finalement, nous montrons que d'autres filtrages, permettant de propager des incohérences locales partielles, peuvent être définis.

Introduction

Relativement à d'autres formalismes pour l'optimisation combinatoire, le formalisme CSP est surtout caractérisé par l'emploi systématique de propriétés de cohérence locale et des algorithmes de filtrage, en particulier la cohérence d'arc.

La notion de cohérence locale peut être caractérisée par un ensemble de propriétés désirables : une cohérence locale est une relaxation de la cohérence dans le sens où tout CSP cohérent admet un CSP équivalent non vide et localement cohérent. La recherche de ce CSP équivalent, qui est unique, défini un problème polynomial résolu par l'algorithme de filtrage associé. De nombreux travaux ont tentés d'étendre la notion de cohérence d'arc à des formalismes autorisant la violation des contraintes. Le premier résultat dans ce sens est probablement [RHZ76] qui étend la cohérence d'arc aux CSP flous (max-min).

Cette extension était relativement simple et l'on pouvait penser que cela serait aussi le cas pour d'autres problèmes similaires tels que le MAX-CSP, introduit dans [SH81, FW92], et dans lequel le but est de minimiser le nombre (ou une somme pondérée) de contraintes violées. Cela n'a pas été le cas. Les travaux suivants ont tenté d'étendre la cohérence de façon systématique en utilisant des cadres axiomatiques caractérisant les propriétés de l'opérateur utilisé pour combiner les violations :

– le cadre des *Semi-Ring CSP* (SCSP) a été introduit dans [BMR95]. Dans ces travaux, l'extension de la cohérence d'arc est induite par une généralisation des opérateurs relationnels fondamentaux tels que la projection, l'intersection et la jointure. La conclusion essentielle est que la cohérence d'arc ainsi étendue fonctionne si l'opérateur de combinaison des violations est idempotent. Ceci inclut le cas des CSP flous (max-min) et quelques autres cas de structures partiellement ordonnées. Dans le cas des MAX-CSP et de problèmes similaires, l'algorithme de filtrage peut ne pas se terminer ou retourne un problème non équivalent.
– Le cadre des CSP valués a été introduit dans [SFV95]. Ici, l'extension de la cohérence d'arc est basée sur la notion de relaxation. Dans le cas d'un opérateur idempotent, les mêmes conclusions que pour les SCSP sont obtenues. Pour le cadre MAX-CSP, on montre que le problème de vérification de la cohérence d'arc étendue est NP-complet.

Simultanément, différent travaux [Wal94, AB98, LMSV98, LMS99] ont tenté de définir des minorants améliorés pour la résolution du MAX-CSP. Ces travaux renoncent à l'extension de la cohérence d'arc pour se limiter au service le plus important : le calcul d'un minorant du coût d'un solution optimale.

Globalement, chacun de ces travaux viole quelques unes des propriétés désirables d'une propriété de cohérence locale. Dans ce travail, nous allons montrer qu'il est possible, via l'ajout d'un seul axiome au cadre des CSP valués, de définir une notion de cohérence d'arc étendue qui conserve toutes les propriétés de la cohérence d'arc classique à l'exception de l'unicité de la fermeture arc cohérente. Cette extension a la propriété sympathique de se réduire aux définitions existantes lorsque l'opérateur de combinaison est idempotent et l'unicité de la fermeture est retrouvée dans ces cas. Chaque fermeture induit un minorant qui généralise et améliore les minorants existants [Wal94, AB98, LMSV98, LMS99]. Finalement, nous considérons des algorithmes de filtrage par arc cohérence plus forts, cherchant à optimiser le minorant défini en propageant des incohérences partielles et non seulement des effacements de valeurs comme dans [LMSV98, LMS99].

1 Notations et définitions

Un problème de satisfaction de contraintes (CSP) est un triplet $\langle X, D, C \rangle$. X est un ensemble de n variables $X = \{1, \ldots, n\}$. Chaque variable $i \in X$ a un domaine de valeurs $d_i \in D$ et peut recevoir toute valeur $a \in d_i$, notée également (i, a). C est un ensemble de contraintes. Chaque contrainte $c \in C$ est définie sur un ensemble de variables $X_c \subset X$ par un sous-ensemble du produit cartésien $\prod_{i \in X_c} d_i$ donnant la liste des combinaisons de valeurs (tuples) autorisées. La cardinalité $|X_c|$ est l'arité de la contrainte c. k dénotera la plus grande arité du CSP. Nous supposons, sans perte de généralité, qu'au plus une contrainte existe sur un ensemble de variables donné. L'ensemble C est partitionné est deux sous-ensembles $C = C^1 \cup C^+$ où C^1 contient toutes les contraintes unaires. La contrainte unaire sur la variable i sera notée c_i. La projection d'un tuple de valeurs t sur un ensemble de variables $V \subset X$ est notée $t_{\downarrow V}$. Cette notation est étendue aux contraintes : la projection $c_{\downarrow V}$ d'une contrainte c sur un ensemble de variables $V \subset X_c$ est la contrainte définie sur V et égale à l'ensemble des projections sur V des tuples de c. Un tuple $t \in c$ tel que $t_{\downarrow \{i\}} = (a)$ est un support pour la valeur (i, a) sur c. Un tuple de valeurs t satisfait une contrainte c si $t_{\downarrow X_c} \in c$. Enfin, un tuple de valeurs sur X est une solution ssi il satisfait toutes les contraintes du CSP.

1.1 CSP valués

Les CSP valués (VCSP) ont été initialement introduits dans [SFV95]. Un CSP valué est obtenu en associant un objet mathématique appelé valuation à chaque contrainte d'un CSP classique. L'ensemble E des valuations est supposé complètement ordonné et son élément maximum est associé aux contraintes dures (inviolables). Quand une instanciation viole un ensemble de contraintes, on calcule sa valuation en combinant les valuations de ces contraintes en utilisant un opérateur dédié noté $\oplus$. Cet opérateur doit satisfaire certaines propriétés capturées par un ensemble d'axiomes définissant une « structure de valuation ».

DÉFINITION 1 *Une* structure de valuation *est définie par un triplet* $\langle E, \oplus, \succcurlyeq \rangle$ *tel que :*

- E *est un ensemble formé de valuations qui est totalement ordonné par* $\succcurlyeq$ *avec un élément maximum noté* $\top$ *et un élément minimum noté* $\perp$ *;*
- $\oplus$ *est un opération binaire interne, commutative et associative sur* E *qui vérifie :*
  - Identité *:* $\forall a \in E, a \oplus \perp = a$ *;*
  - Monotonie *:* $\forall a, b, c \in E, (a \succcurlyeq b) \Rightarrow \big((a \oplus c) \succcurlyeq (b \oplus c)\big)$ *;*
  - Élément absorbant *:* $\forall a \in E, (a \oplus \top) = \top$.

Cette structure de monoïde totalement ordonné commutatif équipé d'un opérateur monotone est classique en raisonnement dans l'incertain[1]. Dans le reste de ce travail, nous supposons que le calcul de $\succ$ et $\oplus$ est toujours polynomial dans la taille des arguments et que E est dénombrable.

Il est maintenant possible de définir les CSP valués. Pour des raisons de généralité, plutôt que de considérer que les valuations sont associées aux contraintes, comme dans [SFV95], nous supposons que les valuations sont associées aux tuples de chacune des contraintes. Comme il a été observé dans [BFM$^+$99], les deux approches sont essentiellement équivalentes.

DÉFINITION 2 *Un CSP valué est un quadruplet* $\langle X, D, C, S \rangle$ *où* $S = \langle E, \oplus, \succcurlyeq \rangle$ *est une structure de valuation,* X *est un ensemble de* n *variables* $X = \{1, \ldots, n\}$. *Chaque variable* $i \in X$ *dispose d'un domaine associé* $d_i \in D$. $C = C^1 \cup C^+$ *est un ensemble de contraintes. Chaque contrainte* $c \in C$ *est définie sur un ensemble de variables* $X_c \subset X$ *comme une fonction de* $\prod_{i \in X_c} d_i$ *vers* E.

Une instanciation t d'un ensemble de variables $V \subset X$ peut être évaluée en combinant, pour chacune des contraintes, les valuations des tuples utilisés dans ces contraintes :

DÉFINITION 3 *Dans un VCSP* $\mathcal{P} = \langle X, D, C, S \rangle$, *la valuation d'une instanciation* t *de* $V \subset X$ *est définie par :*

$$\mathcal{V}_{\mathcal{P}}(t) = \bigoplus_{c \in C, X_c \subset V} [c(t_{\downarrow X_c})]$$

Le problème habituellement considéré est la recherche d'une instanciation complète de valuation minimum. Globalement, la sémantique d'un VCSP est une distribution de valuations sur les instanciations de X. Le choix des axiomes est assez naturel et est habituel en raisonnement dans l'incertain. L'ensemble ordonné E permet d'exprimer différents impacts de violation. La commutativité et l'associativité garantissent que la valuation d'une instanciation ne dépend pas de l'ordre dans lequel les valuations sont combinées. La monotonie de $\oplus$ garantit que les valuations d'instanciations ne peuvent décroître quand les violations de contraintes deviennent plus importantes. Pour une analyse plus détaillée, nous invitons le lecteur à lire [SFV95, LMS99] qui met aussi en évidence la différence entre les opérateurs $\oplus$ qui sont idempotents et ceux qui sont strictement monotones.

[1] E étant restreint à $[0, 1]$, on appelle une tell structure une « co-norme triangulaire » [DP82]

DÉFINITION 4 *Un opérateur* $\oplus$ *est idempotent si* $\forall v \in E, (v \oplus v) = v$. *Il est strictement monotone si* $\forall a, b, c \in E$, *t.q.* $(a \succ c) \wedge (b \neq \top)$, *on a* $(a \oplus b) \succ (c \oplus b)$

Comme le montre [SFV95], ces deux propriétés sont incompatibles dès que $|E| > 2$. Les seules structures de valuation avec un opérateur idempotent correspondent aux CSP classiques et flous qui utilisent $\oplus = \max$. Les autres cadres tels que les MAX-CSP, les CSP lexicographiques ou probabilistes utilisent un opérateur strictement monotone.

Le filtrage par cohérence d'arc doit fournit un problème équivalent au problème d'origine. Plusieurs notions d'équivalences entre CSP valués ont été définies dans [SFV95, LMS99]. Elles permettent de comparer des VCSP qui utilisent des structures de valuation distinctes. Ici, nous ne comparerons que des VCSP qui partage la même structure de valuation et nous pouvons simplifier et renforcer la notion de problèmes équivalents :

DÉFINITION 5 *Deux VCSP* $\mathcal{P} = \langle X, D, C, S\rangle$ *et* $\mathcal{P}' = \langle X, D, C', S\rangle$ *sont équivalents ssi pour toute instanciation complète* t, *on a :*

$$\mathcal{V}_{\mathcal{P}}(t) = \mathcal{V}_{\mathcal{P}'}(t)$$

2 La cohérence d'arc revisitée

Dans cette section, nous donnons une définition inhabituelle et très naïve de la cohérence d'arc classique. Elle permettra une extension plus facile au cas valué. Nous considérons par la suite que les domaines finaux (arc cohérents) sont définis par les contraintes unaires $c_i \in C^1$.

DÉFINITION 6 *Un CSP classique est arc cohérent si pour toute contrainte* $c \in C^+$ *et pour toute variable* $i \in X_c$, *on a* $(c_i \subset c_{\Downarrow\{i\}})$ *et* $(c_i \supset c_{\Downarrow\{i\}})$.

Habituellement, un CSP est dit arc cohérent dès lors que $(c_i \subset c_{\Downarrow\{i\}})$. Notre définition rend explicite l'hypothèse implicite que les tuples qui utilisent une valeur effacée sont retirées des contraintes *i.e.*, que $(c_i \supset c_{\Downarrow\{i\}})$. Si ce n'est pour le fait qu'un CSP avec des domaines vides peut être arc cohérent avec cette définition, elle est équivalente aux définitions habituelles (nous considérons que la possible vacuité de la fermeture arc cohérente est une propriété séparée). La première condition garantit que chaque valeur a un support et la seconde que les tuples sont tous composés de valeurs viables (ayant un support). Globalement, chaque valeur a ainsi un support viable.

Un algorithme de filtrage peut ainsi être défini par l'application itéré jusqu'à stabilisation de deux procédures, chacune établissant une des propriétés ci-dessus : Projection établit le fait que $(c_i \subset c_{\Downarrow\{i\}})$ en effaçant les valeurs sans support. Extension établit le fait que $(c_i \supset c_{\Downarrow\{i\}})$ en effaçant les tuples contenant des valeurs effacées. Les contraintes unaires c_i sont utilisées pour noter les valeurs effacées. La terminaison découle du fait que chaque itération d'AC, si ce n'est la dernière, efface au moins un tuple ou une valeur et il n'y en a que $O(e.d^k)$. Dans le cas binaire, Projection et Extension sont en $O(d)$ et chaque itération d'AC est en $O(e.d^2)$. Cela montre que notre algorithme de filtrage par cohérence d'arc naïf est en $O(e^2.d^4)$.

AC retourne un CSP qui est équivalent au CSP original puisque Projection et Extension retire seulement des valeurs et des tuples qui ne peuvent participer à une solution. Le CSP final est arc cohérent puisqu'il est clos sous Projection and Extension. La mauvaise complexité temporelle de cet algorithme naïf, comparée à la complexité optimale en $O(e.d^2)$ obtenue par AC-4 sur les CSP binaires doit être vue comme la conséquence de sa simplicité voulue : le but de ce papier est de montrer que la cohérence d'arc peut être étendue e.g. aux MAX-CSP. Nous avons choisi de décrire

Algorithme 1 – Filtrage par cohérence d'arc naïf

```
Projection(c, i, a);
m ← 0;
si ∄t ∈ c t.q. t↓{i} = (a) alors
    Effacer (a) de c_i;
    m ← 1;
retourner m ;

Extension(i, a, c);
m ← 0;
si (a) ∉ c_i alors
    pour chaque t ∈ c t.q. t↓{i} = (a) faire
        Effacer t de c;
        m ← 1;
retourner m ;

AC(X, D, C^1 ∪ C^+);
m ← 1;
tant que (m) faire
    m_c ← m_i ← 0;
    pour chaque c ∈ C^+ faire
        pour chaque i ∈ X_c faire
            pour chaque a ∈ d_i faire m_c ← (m_c ∨ Projection(c, i, a));
    pour chaque i ∈ X faire
        pour chaque c ∈ C^+ t.q. i ∈ X_c faire
            pour chaque a ∈ d_i faire m_i = (m_i ∨ Extension(i, a, c));
    m ← m_i ∨ m_c ;
```

un algorithme de filtrage dont l'extension sera simple. Pour cela, il faudra essentiellement garantir que les opérations élémentaires Projection et Extension étendues préservent la sémantique du problème.

2.1 Cohérences d'arc pour contraintes molles

Nous commençons par un exemple introductif. Dans le reste du papier, afin d'illustrer les notions introduites sur des exemples « concrets », nous considérerons des MAX-CSP binaires correspondant à des CSP valués utilisant le structure de valuation strictement monotone $\langle \mathbb{N} \cup \{+\infty\}, +, \geq \rangle$. Pour décrire de tels problèmes, nous utilisons une représentation à base de graphes où les sommets représentent les valeurs et des arêtes pondérées représentent les paires de valeurs des contraintes avec leur valuation associée. Dans cette représentation, nous omettrons les arêtes pondérées par 0 ($\perp$) et tous les poids égaux à $+\infty$ ($\top$). Les contraintes unaires sont représentées par des poids associés aux sommets, les poids égaux à 0 étant omis.

Considérons le MAX-CSP de la figure 1(a). Il a deux variables notées 1 et 2, chacune ayant deux valeurs a et b et une seule contrainte. La contrainte interdit les paires $((1, b), (2, b))$ avec un coût de 1, interdit les paires $((1, a), (2, a))$ et $((1, b), (2, a))$ complètement (avec un coût infini). La paire $((1, a), (2, b))$ est autorisée et l'arête correspondante est donc omise.

Si on affecte la valeur b à la variable 1, on sait qu'un coût de 1 au moins devra être payée car le meilleur support pour $(1, b)$ a un coût de 1. Un opérateur de projection pourrait rendre ce fait explicite et induire une contrainte unaire sur 1 qui interdit $(1, b)$ avec un coût de 1. Cependant, si l'on se contente d'ajouter cette contrainte au problème, comme cela a été initialement proposé dans [BMR95], le problème obtenu n'est pas équivalent. L'instanciation complète $((1, b), (2, b))$ qui avait initialement un coût de 1 a maintenant un coût de 2. Afin de préserver l'équivalence, nous devons "compenser" l'ajout de cette contrainte unaire. Pour cela, on soustrait simplement 1 des valuations des tuples de la contrainte projetée qui contiennent la valeur $(1, b)$. Le problème

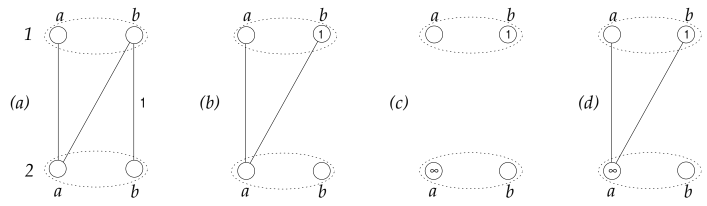

FIG. 1 – Quatre MAX-CSP équivalents

équivalent est décrit dans la figure 1(b) : l'arête $((1, b), (2, b))$ de coût 1 a disparue (le poids associé est nul) alors que l'arête $((1, b), (2, a))$ est inchangée puisqu'elle est de poids infini. On peut répéter ce processus pour la variable 2 : tous les supports de la valeur $(2, a)$ ont un coût infini. On peut donc ajouter une contrainte unaire qui interdit complètement la valeur $(2, a)$. Dans ce cas précis, et parce que la valuation $+\infty$ est telle que $\forall v \in E, +\infty \oplus v = +\infty$, on peut soit compenser cette ajout (figure 1(c)) ou l'ignorer (figure 1(d)). Dans les deux cas, un MAX-CSP équivalent est obtenu. En fait, le poids associé aux arêtes incidentes au sommet $(2, a)$ ne peut affecter la sémantique du problème puisque cette valeur est totalement interdite. Le problème 1(c) est intéressant car il est complètement résolu. Cependant, nous préférons le problème 1(d) car il explicite les contraintes induites au niveau unaire.

2.2 Un nouvel axiome pour les VCSP

Pour généraliser cette idée à d'autres structures de valuations, il faut pouvoir « compenser » l'information ajoutée par la projection. Ceci est rendu possible via l'introduction d'un nouvel axiome au cadre VCSP :

DÉFINITION 7 *Une structure de valuation* $\langle E, \oplus, \succcurlyeq \rangle$ *est* juste *si pour toute paire de valuations* $u, v \in E$, $v \preccurlyeq u$, *il existe une valuation* $w \in E$ *telle que* $v \oplus w = u$. w *sera appelée une différence de* u *et* v. *L'unique différence maximum de* u *et* v *sera notée* $u \ominus v$.

Ce simple axiome est satisfait par la grande majorité des formalismes existants, qu'ils s'appuient sur des opérateurs strictement monotones ou idempotents.

- les CSP classiques peuvent être définis comme des VCSP utilisant l'ensemble de valuations $E = \{0, 1\}$, où $0 = \bot \prec 1 = \top$. L'opérateur de combinaison est $\max$ et l'on a $0 \ominus 0 = 0$, $1 \ominus 0 = 1$ et $1 \ominus 1 = 1$. Ainsi, $\ominus = \max$ aussi. Remarquez que 0 est une autre différence entre 1 et 1, mais elle n'est pas maximum.
- les CSP possibilistes, qui définissent un dual min-max des CSP flous (max-min) peuvent être définis en utilisant l'ensemble de valuations $\mathbb{N} \cup \{+\infty\}$, ordonné comme d'habitude et en utilisant $\oplus = \max$. Ici encore, une différence existe et on montre que $\ominus = \max$.
- Les MAX-CSP peuvent êtres définis en utilisant l'ensemble de valuations $\mathbb{N} \cup \{+\infty\}$, ordonné comme d'habitude et en utilisant $\oplus = +$. Ici encore, une différence existe et on montre que $u \ominus v = u - v$ pour des valuations finies. Sinon, pour toute $v \in E$, on a $+\infty \ominus v = +\infty$.

Il faut noter que d'autres formalismes tels que celui des CSP probabilistes [FL93] ou lexicographiques [FLS93] peuvent être simplement réduits en MAX-CSP comme cela a été montré dans [SFV95] et peuvent donc profiter des résultats de ce papier. Ceci est particulièrement important pour les CSP lexicographiques pour lesquels une différence n'est pas toujours définie.

PROPRIÉTÉ 1 *Soit* $S = \langle E, \oplus, \succcurlyeq \rangle$ *une structure de valuation juste. Alors* $\forall u, v, w, \in E, w \preccurlyeq v$, *on a* $(v \ominus w) \preccurlyeq v$ *et* $(u \oplus w) \oplus (v \ominus w) = (u \oplus v)$.

Preuve. *Par définition,* $(v \ominus w) \oplus w = v$. *Du fait de la monotonie de* $\oplus$, *cela prouve que* $(v \ominus w) \preccurlyeq v$ *(cette inégalité devient stricte si* $\oplus$ *est strictement monotone et* $v \neq \top$*). La seconde propriété découle de la commutativité et de l'associativité de* $\oplus$ *: on a* $(u \oplus w) \oplus (v \ominus w) = u \oplus ((v \ominus w) \oplus w) = (u \oplus v)$. □

2.3 Filtrage par cohérence d'arc molle

Il est maintenant possible de définir une version étendue du filtrage par cohérence d'arc basée sur l'algorithme précédant. Notre but est définir un algorithme générique qui fonctionne sur toutes les structures de valuation justes. On rappelle que c_i dénote la contrainte unaire portant sur la variable i. Ainsi, $c_i(a)$ est la valuation associée à la valeur (i, a) dans la contrainte unaire c_i. De la même façon que l'on efface des valeurs lors d'un filtrage classique, nous utiliserons des expressions telles que $c(t) \leftarrow v$ pour indiquer que la valuation associée avec le tuple t dans c est fixée à v. Nous utiliserons également les définitions suivantes :

DÉFINITION 8 *Soit* $v, w \in E$, *deux valuations. Si* $v \neq \bot$ *et* $v \ominus v = v$, *nous dirons que* v *est* absorbante. *Si* $v \oplus w \neq w$ *nous dirons que* v affecte w.

Nous supposons que ces deux propriétés peuvent être testées en temps constant (c'est le cas dans les structures de valuations déjà considérées). Notez que $\top$ est absorbante dans toutes les structures de valuation. Quand $\oplus$ est idempotent, toutes les valuations, sauf $\bot$, sont absorbantes. Nous sommes maintenant prêts à étendre les deux procédures Projection and Extension.

Algorithme 2 – Procédures étendues pour la cohérence d'arc molle

Projection(c, i, a);
$m \leftarrow 0$;
$v \leftarrow \top$;
pour chaque $t \in c$ t.q. $t_{\Downarrow\{i\}} = (a)$ **faire**
  $v \leftarrow \min(v, c(t))$;
si v affecte $c_i(a)$ **alors**
  $v' \leftarrow$ Choose(v);
  $c_i(a) \leftarrow c_i(a) \oplus v'$;
  $m \leftarrow 1$;
  pour chaque $t \in c$ t.q. $t_{\Downarrow\{i\}} = (a)$ **faire**
    $c(t) \leftarrow c(t) \ominus v'$;
retourner m ;

Extension(i, a, c);
$m \leftarrow 0$;
$v \leftarrow c_i(a)$;
si v est absorbante **alors**
  $v' \leftarrow$ Choose(v);
  pour chaque $t \in c$ t.q. $t_{\Downarrow\{i\}} = (a)$ **faire**
    si v' affecte $c(t)$ **alors**
      $c(t) \leftarrow c(t) \oplus v'$;
      $m \leftarrow 1$;
1 $c_i(a) \leftarrow c_i(a) \ominus v'$;
retourner m ;

Analysons ces procédures. Elles utilisent toutes les deux la fonction Choose(v) qui prend comme argument une valuation et retourne une valuation. On supposera par la suite que Choose(v) est l'identité. Dans Projection, on cherche un support de valuation minimum pour la valeur (i, a) sur la contrainte c et on conserve sa valuation dans v. Si (i, a) est utilisée dans une instanciation, il faudra au moins payer cette valuation. Cela défini une contrainte unaire possible. Avant d'ajouter cette contrainte, on vérifie si elle modifie le problème : on teste si v affecte la valuation $c_i(a)$ existante au niveau unaire. Si c'est le cas, on ajoute la contrainte unaire induite, on note que le problème est modifié et on compense l'ajout de contrainte en soustrayant v des valuations de touts les supports de (i, a) dans la contrainte. Ceci est toujours possible car la structure de valuation est juste et par construction v est inférieure aux valuations des supports.

Dans Extension, on vérifie que la valuation v de la valeur a est absorbante. Cela est nécessaire pour garantir le terminaison (voir section 4). Ensuite, pour chacun des supports de (i, a) sur c, on vérifie que v affecte la valuation du support et si c'est le cas, on la modifie. La ligne 1 compense cette modification au niveau unaire. Cette ligne est inutile puisque v est absorbante mais elle sera nécessaire par la suite dans une seconde version d'Extension dans laquelle v pourra être non absorbante.

THÉORÈME 1 *Étant donné un VCSP juste $P = \langle X, D, C, S\rangle$, $i \in X$, $a \in d_i$ et $c \in C$ t.q. $i \in X_c$, l'application de Projection(c, i, a) ou Extension(i, a, c) à P retourne un VCSP équivalent.*

Preuve. *Considérons une instanciation complète t. Si $t_{\Downarrow\{i\}} \neq (a)$ alors la valuation de l'instanciation est inchangée par l'application de Projection ou Extension car ces procédures ne modifie que les valuations de tuples qui contiennent (i, a). Si $t_{\Downarrow\{i\}} = (a)$, notons $\alpha = c_i(a)$ et $\beta = c(t_{\downarrow X_c})$. La valuation initiale de t peut s'écrire $\omega \oplus \alpha \oplus \beta$ où ω contient les valuations combinées de toutes les contraintes hormis c et c_i. Après l'application de Projection, la valuation de t devient $\omega \oplus (\alpha \oplus v) \oplus (\beta \ominus v) = \omega \oplus \alpha \oplus \beta$ (propriété 1) et est donc inchangée. De la même façon, l'application de Extension aboutit à une valuation de $\omega \oplus (\alpha \ominus v) \oplus (\beta \oplus v) = \omega \oplus \alpha \oplus \beta$, inchangée (propriété 1).* □

Considérons l'algorithme AC initial utilisant les nouvelles procédures Projection and Extension. S'il se termine, il retourne nécessairement un problème équivalent. La terminaison reste à prouver dans le cas général mais elle est simple à montrer dans les cas les plus importants en pratique *i.e.*, quand $\oplus$ est idempotent ou strictement monotone :

- Cas idempotent : $\oplus$ est égal à $\max$ (voir [SFV95]) et évidemment $\ominus = \max$ aussi. Le nombre de valuations différentes que l'on peut obtenir en combinant des valuations du problème avec $\max$ est fini et égal au nombre de valuations différentes qui existe dans le problème. Chaque application de Projection ou Extension ne peut que faire croître des valuations dans les contraintes. Le nombre de valuations possibles étant fini, l'algorithme se termine.
- Cas strictement monotone : la seule valuation absorbante est $\top$. Pour toutes les autres valuations $v \neq \top$, on a $v \ominus x \prec v$ pour tout $x \neq \bot$ (cela découle de la monotonie stricte). Si l'on considère les valuations des tuples des contraintes, chaque appel à Projection ne peut que réduire les valuations alors que chaque appel à Extension ne peut que fixer des valuations à $\top$, élément maximum et absorbant qui ne peut décroître. Globalement, chaque itération de l'algorithme AC peut soit diminuer des valuations ou les fixer à $\top$. Dans le premier cas, la valuation ne peut descendre en dessous de $\bot$ et dans le second elle ne peut plus changer. La terminaison découle du fait qu'il n'y a qu'un nombre fini de tuples dans les contraintes. Notez que cette preuve reste valide si Choose retourne une valeur inférieure ou égale à v et strictement plus grande que $\bot$.

Nous disposons donc d'un algorithme de filtrage étendu qui se termine et retourne un problème équivalent pour toutes structure de valuation juste ayant un opérateur de combinaison $\oplus$ idempotent ou strictement monotone. La spécialisation au cas classique de cet algorithme est identique à notre algorithme de filtrage classique. Il est maintenant possible de définir la propriété de cohérence d'arc molle :

DÉFINITION 9 *Un VCSP juste est arc cohérent si pour toute contrainte $c \in C^+$, pour toute variable $i \in X_c$, pour toute valeur $a \in d_i$ on a :*

- *la valuation minimum des supports de (i, a) sur c n'affecte pas $c_i(a)$.*
- *si $c_i(a)$ est absorbante, alors elle n'affecte aucune des valuations des supports de (i, a) sur c.*

Notre algorithme de filtrage est correct par définition de la propriété de cohérence d'arc qui correspond précisément aux propriétés établies par Projection et Extension. Comme pour le filtrage, la spécialisation de la définition au cas classique correspond à la définition classique de la cohérence d'arc.

Il reste à prouver que l'algorithme est polynomial. Là encore, aucune preuve générique n'a été établie, nous avons préférer nous limiter au cas le plus intéressant des MAX-CSP binaires. Dans ce cas, Projection et Extension sont en $O(d)$. Dans AC, chaque itération effectue $O(ed)$ appels à ces procédures. Chaque itération, si ce n'est la dernière, efface une valeur ou fixe une valuation à $0 = \bot$ ou à $+\infty = \top$. Il n'y a que $e.d^2$ tuples et $n.d$ valeurs donc cet algorithme naïf est en $O(e^2.d^4)$ et est effectivement polynomial. Naturellement, il sera indispensable de transformer ce résultat théorique en un algorithme efficace en temps et en espace.

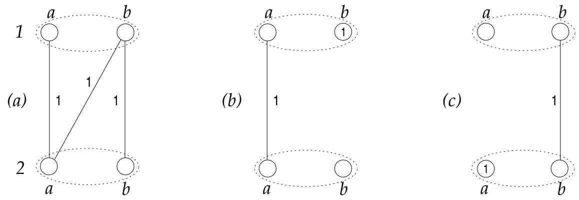

FIG. 2 – Un MAX-CSP et deux fermetures arc cohérentes distinctes mais équivalentes

Enfin, nous observons que la confluence du filtrage est perdue : la fermeture arc cohérente d'un problème n'est pas unique comme le montre la figure 2. Le MAX-CSP considéré est similaire au problème de la figure 1 si ce n'est que les tuples interdits ont un coût de 1. Si l'on commence le filtrage par la variable 1, on détecte que $(1, b)$ a support minimum de coût 1 et l'on obtient le problème 2(b). Ce réseau est arc cohérent, équivalent au problème 2(a). Si l'on commence par la variable 2, on note que $(2, a)$ a un support minimum de coût 1 et l'on obtient le problème 2(c). Ce réseau est arc cohérent, équivalent au réseau 2(a) mais est distinct du problème 2(b).

3 Minorants induits

Dans le cadre VCSP, comme dans le cadre CSP, chaque variable doit recevoir une valeur. Donc, la valuation suivante est toujours un minorant du la valuation d'une solution optimale :

$$\ell = \bigoplus_{i \in X} [\min_{a \in d_i} c_i(a)]$$

Calculée après filtrage par cohérence d'arc, ce minorant sera noté ℓ_{ac}. Il dépend de la fermeture arc cohérente trouvée qui dépend de l'ordre dans lequel contraintes et variables sont traitées durant le filtrage. Un VCSP est dit *maximalement arc cohérent* s'il est arc cohérent et si ℓ_{ac} est maximum sur tous les problèmes arc cohérents équivalents. Le problème d'établir l'arc cohérence maximum est certainement important, mais il y a quelques raisons de penser qu'il s'agit d'un problème NP-dur (voir [Sch98]). C'est un travail qui reste à faire.

Nous comparons ℓ_{ac} à des minorants récents pour les MAX-CSP binaires. Comme exemple, considérons le problème de la figure 3. Il a quatre variables avec deux valeurs chacune et contient quatre contraintes. Le filtrage par cohérence d'arc produit le réseau de droite. Le minorant ℓ_{ac} est ici égal à 4. Considérons d'abord le minorant défini par les *reversible directed arc consistency counts* (RDAC) introduit dans [LMSV98, LMS99]. Il est paramétré par des booléens indiquant une direction par contrainte. Pour chaque contrainte c et pour chaque valeur a de la variable i indiquée par la direction choisie, le compteur $w_{i,a}$ est incrémenté si la valeur n'a pas de support sur c. Le minorant induit est la somme de minima des w_{ia} par variable. Si l'on filtre par cohérence d'arc et que l'on examine les contraintes dans le même ordre que RDAC en commençant toujours par la variable que RDAC examine, il est clair que Projection découvrira exactement les mêmes absences de support que RDAC et induira les compteurs $w_{i,a}$ sous forme de contraintes unaires. De plus, il examinera l'autre coté de chacune des contraintes et inférera éventuellement plus de contraintes unaires, fournissant un minorant au moins aussi fort. Dans notre exemple, une valeur de chaque domaine a un support de coût nul. RDAC ne peut donc inférer que 4 compteurs de 1 pour les 8 valeurs. Idéalement, si les contraintes sont correctement orientées (e.g. 2 contraintes orientées vers la variable 1 et deux autres vers la variable 3), ces 4 coûts induisent un minorant de 2. Cependant, un minorant inutile de 0 peut aussi être inféré si les contraintes sont mal orientées e.g. si une contrainte est orientée vers chaque variable. Le filtrage par cohérence d'arc est donc sensiblement plus fort.

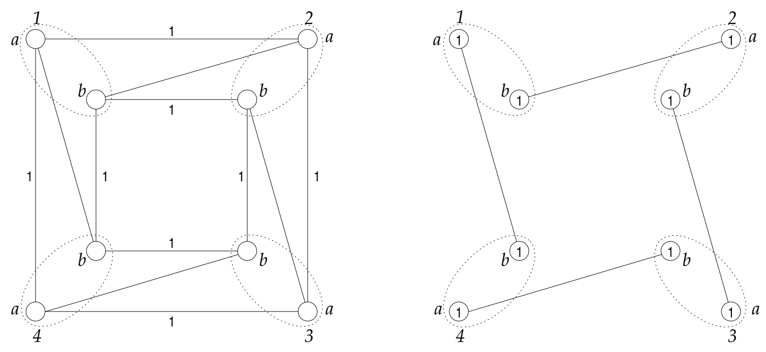

FIG. 3 – Un MAX-CSP et une fermeture arc cohérente

Le second minorant considéré est le minorant basé sur les *weighted arc consistency counts* ou WAC [AB98]). Il est paramétré par un réel α entre 0 et 1 par contrainte. On sait [ABS99] qu'il s'agit d'une généralisation des RDAC. Pour chaque contrainte, les WAC inférés sont exactement égaux à α fois les compteurs qu'aurait inféré RDAC dans une orientation et $(1-\alpha)$ fois ceux qui auraient été inférés dans l'autre direction. Nous pouvons paramétrer le filtrage par cohérence d'arc en modifiant Choose pour qu'il retourne $\alpha.v$ d'un coté et $(1-\alpha).v$ de l'autre et conclure qu'AC peut toujours fournir un minorant au moins aussi fort que celui induit par les WAC : on commence par appliquer Projection une fois sur chaque contrainte/variable en utilisant la procédure Choose paramétrée par la valeur α. Le réseau obtenu présente les même coûts que les WAC, sous forme de contraintes unaires, mais n'est pas nécessairement arc cohérent. Si l'on achève le filtrage en utilisant la version standard de Choose, on ne peut qu'augmenter les coûts et donc améliorer le minorant. Si l'on considère notre exemple, WAC peut idéalement inférer un minorant de 2, comme RDAC, en utilisant e.g. $\alpha = \frac{1}{2}$ pour chaque contrainte.

Notre algorithme de filtrage peut donc fournir des minorants plus forts que les meilleurs minorants connus. De plus il préserve l'équivalence et est défini pour les contraintes non binaires.

4 Propagation des incohérences partielles

Dans cette section nous envisageons la possibilité, dans la procédure Extension, de traiter des valuations non absorbantes. La procédure correspondante est appelée Extension+ et se déduit de Extension en supprimant le test vérifiant que $v = c_i(a)$ est absorbante (voir algorithme 2). La propagation de valuations non absorbantes a été interdite dans la version de base afin de garantir la terminaison : pour des valuations non absorbantes, l'utilisation de Extension+(i, a, c) supprime exactement l'effet d'un appel précédent à Projection(c, i, a) ce qui peut conduire à des boucles infinies. Cependant, la terminaison peut toujours être forcée si elle n'est pas naturelle. Nous trouvons agréable, dans cette section, de considérer le filtrage comme un processus de recherche locale dans l'espace des VCSP équivalents. Le critère optimisé est le minorant ℓ. Un premier voisinage est défini par toutes les applications possibles de Projection et Extension. Son utilisation correspond à l'algorithme de filtrage précédent (algorithme 2) : une recherche gloutonne se termine toujours et le réseau localement optimum atteint est arc cohérent.

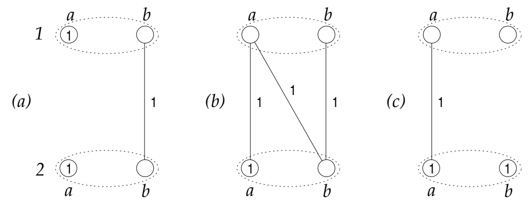

FIG. 4 – La propagation des valuations non absorbantes peut améliorer ℓ_{ac}

Un second voisinage plus large est défini par toutes les applications possibles de Projection et Extension+. Il est particulièrement intéressant dans le contexte e.g. des MAX-CSP où la propagation des valuations non absorbantes peut être la clef pour atteindre une fermeture arc cohérente maximum. Considérons le premier MAX-CSP de la figure 4. Il contient une contrainte binaire qui interdit la paire $((1, b)(2, b))$ et deux contraintes unaires qui interdisent les valeurs$(1, a)$ et $(2, a)$. Le réseau est arc cohérent and le minorant correspondant ℓ_{ac} vaut 0. Si nous appliquons Extension+ à la valeur $(1, a)$, on obtient le réseau équivalent 4(b) qui n'est pas arc cohérent. On peut alors appliquer Projection sur la valeur $(2, b)$ et obtenir le réseau 4(c) qui est arc cohérent avec un minorant $\ell_{ac} = 1$. La propagation de valuations non absorbantes peut permettre le transport de contraintes unaires de variables en variables. Dans certains cas, cela peut être exploité pour augmenter la valuation minimum sur les valeurs $a \in d_i$ de $c_i(a)$ et donc le minorant ℓ_{ac}. Il reste à déterminer s'il est toujours possible d'atteindre une fermeture arc cohérente maximum à partir d'un réseau donné en utilisant ce voisinage étendu. Il est évident que ce n'est pas possible dans tous les cas avec le premier voisinage. À plus long terme, il serait intéressant de déterminer si l'on peut définir un algorithme de filtrage qui exploite Extension+ et qui termine naturellement.

Conclusion

Dans ce travail, nous avons étendu introduit une version étendue de la cohérence d'arc qui s'applique aux CSP classiques et flous, aux MAX-CSP et à d'autres problèmes similaires. Elle étend la notion usuelle de problème arc cohérent équivalent de façon systématique, en l'étendant à des cadres dans lesquels cette notion n'était pas disponible et ce dans le cas de contraintes non nécessairement binaires. Un travail important reste à faire. Avant toute chose, il faut améliorer l'efficacité temporelle et spatiale de l'algorithme de filtrage présenté. L'introduction de la technologie AC-$\{6, 7\}$ dans ce travail devrait faire l'affaire. L'algorithme résultant doit ensuite être injecté dans un algorithme de recherche arborescente similaire à l'algorithme PFC-MRDAC [LMSV98, LMS99] pour la résolution de MAX-CSP. De façon plus large, la technique de « compensation » utilisée dans ce travail devrait permettre l'extension de la plupart (sinon toutes) des cohérences locales classiques aux MAX-CSP et fournir de nouveaux minorants, plus forts.

Remerciements Je remercie sincèrement G. Verfaillie pour sa participation à de stimulantes discussions et M. Lemaître pour ces commentaires sur une première version de ce papier.

Références

[AB98] M. S. Affane and H. Bennaceur. A weighted arc consistency technique for Max-CSP. In *Proc. of the 13th ECAI*, pages 209–213, Brighton, United Kingdom, 1998.

[ABS99] M. S. Affane, H. Bennaceur, and T. Schiex. Comparaison de deux minorants paramétriques pour Max-CSP. In *Actes de JNPC'99*, pages 95–102, 1999.

[BFM$^+$99] S. Bistarelli, H. Fargier, U. Montanari, F. Rossi, T. Schiex, and G. Verfaillie. Semiring-based CSPs and valued CSPs : Frameworks, properties and comparison. *Constraints*, 4 :199–240, 1999.

[BMR95] S. Bistarelli, U. Montanari, and F. Rossi. Constraint solving over semirings. In *Proc. of the 14th IJCAI*, Montréal, Canada, August 1995.

[DP82] D. Dubois and H. Prade. A class of fuzzy measures based on triangular norms. a general framework for the combination of uncertain information. *Int. Journal of Intelligent Systems*, 8(1) :43–61, 1982.

[FL93] H. Fargier and J. Lang. Uncertainty in constraint satisfaction problems : a probabilistic approach. In *Proc. of ECSQARU '93, LNCS 747*, pages 97–104, Grenade, Spain, November 1993.

[FLS93] H. Fargier, J. Lang, and T. Schiex. Selecting preferred solutions in Fuzzy Constraint Satisfaction Problems. In *Proc. of the 1st European Congress on Fuzzy and Intelligent Technologies*, 1993.

[FW92] E.C. Freuder and R.J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58 :21–70, December 1992.

[LMS99] J. Larrosa, P. Meseguer, and T. Schiex. Maintaining reversible DAC for Max-CSP. *Artificial Intelligence*, 107(1) :149–163, January 1999.

[LMSV98] J. Larrosa, P. Meseguer, T. Schiex, and G. Verfaillie. Reversible DAC and other improvements for solving max-CSP. In *Proc. of AAAI'98*, Madison, WI, July 1998.

[RHZ76] A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *IEEE Trans. on Systems, Man, and Cybernetics*, 6(6) :173–184, 1976.

[Sch98] T. Schiex. Maximizing the reversible DAC lower bound in Max-CSP is NP-hard. Technical Report 1998/02, INRA, July 1998.

[SFV95] T. Schiex, H. Fargier, and G. Verfaillie. Valued constraint satisfaction problems : hard and easy problems. In *Proc. of the 14th IJCAI*, pages 631–637, Montréal, Canada, August 1995.

[SH81] L. Shapiro and R. Haralick. Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3 :504–519, 1981.

[Wal94] R. Wallace. Directed arc consistency preprocessing. In M. Meyer, editor, *Selected papers from the ECAI-94 Workshop on Constraint Processing*, number 923 in LNCS, pages 121–137. Springer, Berlin, 1994.