Une comparaison des cohérences d'arc dans les Max-CSP

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

Résumé

Dans [Sch00], une définition de la cohérence d'arc pour les réseaux de contraintes molles (*soft constraints networks*) a été proposée dans les cas précis des réseaux de contraintes valués dits « justes ». Récemment, [Coo02, CS02] ont affiné cette définition dans différentes directions. Ont été ainsi définies 3 notions distinctes : cohérence d'arc, cohérence d'arc directionnelle, cohérence d'arc directionnelle complète.

Contrairement à ce qui existe dans le cas des réseaux de contraintes classiques, la cohérence d'arc directionnelle apparaît comme plus forte que la cohérence d'arc simple car elle satisfait une propriété d'optimalité locale que ne satisfait pas nécessairement la cohérence d'arc.

Dans le but de choisir le type de cohérence à utiliser dans le cadre d'un algorithme de type « séparation-évaluation » ou plus simplement pour un prétraitement du réseau avant application d'un algorithme existant tel que PFC-{M}RDAC [LMS99], nous avons expérimenté ces différentes formes de cohérence locale en comparant les minorants induits par chacune d'entre elles ainsi que celui utilisé dans les algorithmes PFC-{M}RDAC en début de recherche. Il apparaît que la notion d'arc cohérence directionnelle complète est la plus prometteuse des notions produites jusqu'ici.

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. Dans [Sch00], une définition de la cohérence d'arc étendue à une large sous-classe des de réseaux de contraintes valués a été proposée.

Plus récemment, ces travaux ont été poursuivis dans [Coo02, CS02]. En dehors du fait qu'ils prouvent que la cohérence d'arc peut s'établir en temps $O(ed^2)$ et en espace $O(ed)$ dans le cas de structures de valuation strictement monotones (telles que les MAX-CSP), ces travaux ont introduit deux nouveaux types de cohérence d'arc : la cohérence d'arc directionnelle (DAC, *directed arc consistency*) et la cohérence d'arc directionnelle complète (FDAC, *full directed arc consistency*) qui correspond à la conjonction de la cohérence d'arc et de la cohérence d'arc directionnelle. Comme la cohérence d'arc, ces propriétés peuvent s'établir en temps $O(ed^2)$ et en espace $O(ed)$ dans le cas de structures de valuations strictement monotones.

D'un point de vue théorique, [CS02] montre que la cohérence d'arc directionnelle satisfait une propriété d'optimalité locale en terme de qualité du minorant produit appelée *irréductibilité d'arc* qui n'est pas nécessairement satisfaite par la cohérence d'arc et que la cohérence d'arc directionnelle permet de résoudre les CSP valués justes structurés en arbre.

De façon similaire, il avait été montré dans [Sch00] que la cohérence d'arc permettait de produire des minorants plus forts que ceux produits par les *reversible DAC* utilisés dans les algorithmes PFC-{M}RDAC [LMS99].

Nous avons donc comparé en pratique la force des minorants produits par ces différentes formes de cohérences locales qui pourront être utilisées aussi bien dans le cadre d'un algorithme de type « séparation-évaluation » que comme prétraitement avant l'utilisation d'un autre algorithme de résolution (systématique ou non).

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 $X = \{1, \dots, n\}$. 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 trois sous-ensembles $C = C^0 \cup C^1 \cup C^+$ où C^0 contient une unique contrainte d'arité nulle, C^1 contient toutes les contraintes unaires et C^+ contient les autres contraintes. La contrainte d'arité nulle sera noté $c_\varnothing$. La contrainte unaire sur la variable i sera notée c_i. Si $J \subseteq X$ est un ensemble de variables, alors $\ell(J)$ est défini comme l'ensemble des affectations possibles de J *i.e.*, le produit cartésien $\prod_{i \in J} d_i$ des domaines des variables de J. 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* $\bot$ *;*
- $\oplus$ *est un opération binaire interne, commutative et associative sur* E *qui vérifie :*
  - Identité *:* $\forall a \in E, a \oplus \bot = a$ *;*

– 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 $\oplus$ peut se faire en temps constant.

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.

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.

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 :*

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

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

2 La cohérence d'arc et d'arc directionnelle

Les opérations de base d'établissement de la cohérence d'arc introduites dans [Sch00, CS02] nécessitent de pouvoir déplacer les valuations d'une contrainte à une autre et pour cela de disposer d'un moyen de compenser l'ajout d'une valuation α sur une contrainte. À notre connaissance, un tel mécanisme de compensation a été utilisé pour la première fois dans [Kos99] dans le cadre de MAX-CSP. Dans le cadre des VCSP, cela est rendu possible au moyen d'un axiome supplémentaire :

DÉFINITION 6 *Dans une structure de valuation $S = \langle E, \oplus, \succcurlyeq \rangle$, si $u, v \in E, u \succcurlyeq v$ et qu'il existe une valuation w telle que $w \oplus v = u$, alors w est appelé une différence de u et de v. La structure de valuation S est* juste *si pour toute paire de valuations $u, v \in E, u \succcurlyeq v$, il existe une différence maximale de u et de v. Cette unique différence maximale de u et v est notée $u \ominus v$.*

Par le biais de l'utilisation de la différence, il est possible d'appliquer à un réseau de contraintes valués des opération locales qui modifient le réseau en préservant sa sémantique [CS02].

Examinons deux transformations de ce type sur un exemple sur un réseau de type MAX-CSP (utilisant la struture de valuation $\langle \mathbb{N} \cup \{\infty\}, +, \geq \rangle$). Pour décrire un tel problème, nous utilisons un représentation par un grpahe non orienté dans lequel les sommets représentent les valeurs. Pour toute paire de variables $i, j \in X$ telle que $c_{ij} \in C$, pour toute paire de valeurs $a \in d_i, b \in d_j$ telles que $c_{ij}(a, b) \neq \bot = 0$, une arête connecte les valeurs (i, a) et (j, b). Le poids de l'arête et égal a $c_{ij}(a, b)$. Les contraintes unaires sont représentées par des poids associés aux sommets, les poinds nuls étant omis.

Considérons par exemple le MAX-CSP pondéré de la figure 1(a). Il a 2 variables appellées 1 et 2, chacune avec deux valeurs a et b ainsi qu'une unique contrainte. Cette contrainte interdit la paire $((1, b), (2, b))$ avec un coût de 1 et interdit les paires $((1, a), (2, a))$ et $((1, b), (2, a))$ totalement (avec un coût ∞). La paire $((1, a), (2, b))$ is totalement autorisée et il n'y a donc pas d'arête correspondante.

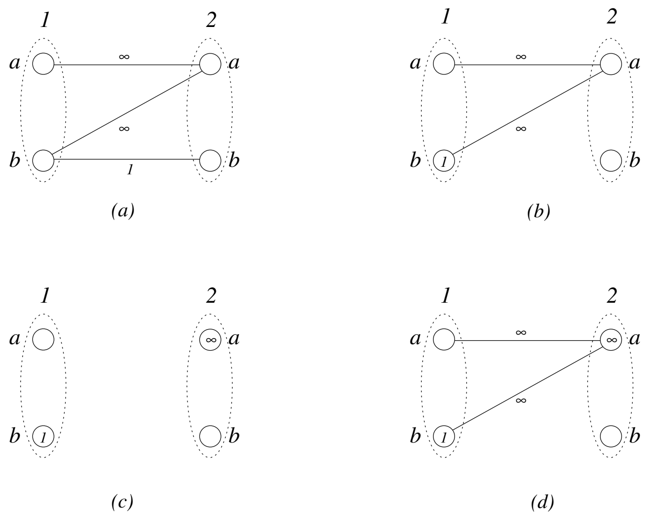


FIG. 1 – Quatre instances équivalentes de MAX-CSP

Si l'on affecte la valeur b à 1, il est certain qu'un coût de 1 doit être payé puisque toute extension de $(1, b)$ à 2 ont un coût de 1 au moins. En projetant ce coût minimum de c_{12} sur 1 en rajoutant une contrainte unaire qui interdit $((1, b)$ avec un coût de 1, on rend ce fait explicite mais on modifie la sémantique du problème. L'affectation $((1, b), (2, b))$ de coût 1 intialement a maintenant un coût de 2. Pour préserver l'équivalence, il suffit de compenser cette addition en soustrayant 1 aux coûts des paires de c_{12} qui contiennent $(1, b)$. On obtient ainsi le réseau de la figure 1(b) : l'arête $((1, b), (2, b))$ de coût 1 a disparue (le poids est maintenant nul) alors que l'arête $((1, b), (2, a))$ est inaffectée puisqu'elle a un coût infini. Un tel processus peut être répété pour la variable 2 car toutes les extensions de $(2, a)$ ont un coût infini et l'on peut donc ajouter une contrainte unaire qui interdit $(2, a)$ avec un coût infini. Dans ce cas précis, et parce que la valuation ∞ vérifie $\infty \oplus \infty = \infty$, on peut soit décider de compenser cette addition (Figure 1(c)) ou non (Figure 1(d)). Dans les 2 cas, problème obtenu est équivalent au problème d'origine. Le problème de la Figure 1(d) est préférable car il rend l'information explicite au niveau unaire et binaire.

De tels opérateurs permettant de modifier un réseau de contrainte molles sans modifier sa sémantique seront appelés des transformations préservant l'équivalence :

DÉFINITION 7 *Le sous-réseau d'un VCSP* $V = \langle X, D, C, S\rangle$ *induit par* $C' \subseteq C$ *est le VCSP* $V(C') = \langle X', D', C', S\rangle$ *où* $X' = \cup_{c \in C'} X_c$ *et* $D' = \cup_{i \in X'} \{d_i\}$.

DÉFINITION 8 *Pour un VCSP* $V = \langle X, D, C, S\rangle$, *et étant donné un ensemble de contraintes* $C' \subseteq C$ *une* transformation préservante *sur* C' *est une opération qui transforme le sous-problème* $V(C')$ *induit par* C' *en un VCSP équivalent.*

Dans le cas où C' ne contient que des contraintes d'arité inférieure à un, on parlera de nœud transformation préservante, si C' ne contient qu'une seule contrainte d'arité supérieure à 1 on parlera d'arc transformation préservante, etc.

Étant donné une contrainte $c_p \in C^+$ et une variable $i \in P$, les deux opérations définies par l'algorithme 1 sont deux exemples d'arc transformation préservantes sur $C' = \{c_P, c_i\}$.

Algorithme 1: Deux arc transformation préservantes

Procedure Project(c_P, i, a)
 $\beta \leftarrow \min_{t \in \ell(P-\{i\})}(c_P(t, a))$;
 $c_i(a) \leftarrow c_i(a) \oplus \beta$;
 pour chaque ($t \in \ell(P - \{i\})$) **faire**
  $c_P(t, a) \leftarrow c_P(t, a) \ominus \beta$;

Procedure Extend(i, a, c_P)
 pour chaque ($t \in \ell(P - \{i\})$) **faire**
  $c_P(t, a) \leftarrow c_P(t, a) \oplus c_i(a)$;
 $c_i(a) \leftarrow c_i(a) \ominus c_i(a)$;

2.1 Cohérence d'arc dans les VCSP strictement monotones binaires

Dans la suite de ce document, nous nous plaçons dans le cas de VCSP binaires. Les définitions et algorithmes dans le cas n-aires sont présentés dans [CS02]. Nous rappelons la définition de l'arc cohérence dans les VCSP justes :

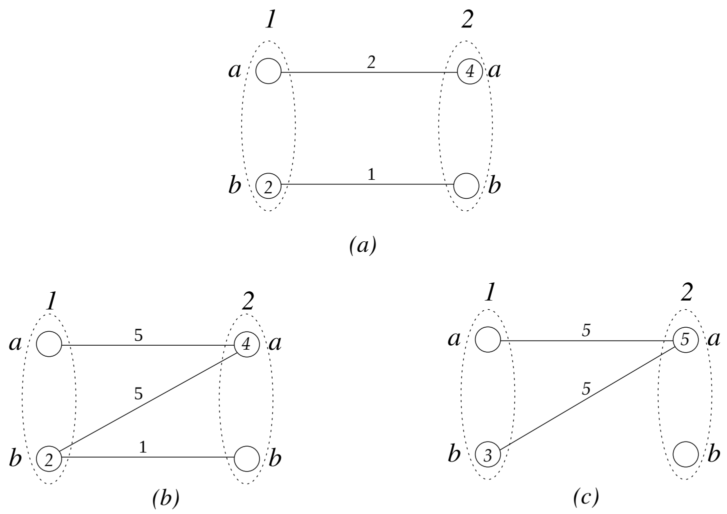


FIG. 2 – (a) Un exemple de VCSP et comment les conditions 1 et 2 sont établies

.

DÉFINITION 9 *Un VCSP juste binaire est* arc cohérent *si pour tout* $i, j \in X$ *tels que* $c_{ij} \in C^+$*, pour tout* $a \in d_i$ *on a :*

1. $\forall b \in d_j, c_{ij}(a,b) = \big(c_i(a) \oplus c_{ij}(a,b) \oplus c_j(b)\big) \ominus \big(c_i(a) \oplus c_j(b)\big)$.
2. $c_i(a) = \min_{b \in d_j}(c_i(a) \oplus c_{ij}(a,b))$

Pour bien comprendre la condition 1 de la définition 9 dans le cas le plus général, considérons une structure valuation dans laquelle les valuations sont prises dans l'ensemble $\{0,1,2,3,4,5\}$ avec $\forall \alpha, \beta \in E, (\alpha \oplus \beta = \min(5, \alpha + \beta))$. $5 = \top$ est absorbant et vérifie $5 \ominus \alpha = 5$ pour tout $\alpha \preccurlyeq 5$. Cette structure de valuation correspond à un cas d'intérêt pratique : celui où l'on est en train de résoudre un problème et que l'on connaît un majorant (ici 5) de l'optimum (par exemple dans le cas d'un algorithme de de type « séparation-évaluation »). La figure 2(a) montre un VCSP à 2 variables sur cette structure. La figure 2(b) montre le résultat de l'établissement de la propriété 1 de la définition 9 qui n'a pas d'équivalent dans le cas classique : les valuations $c_{12}(a,a)$ et $c_{12}(b,a)$peuvent être tous les deux fixées à $5 = \top$, sans aucune compensation au niveau unaire et pourtant sans changer la sémantique du problème. La figure 2(c) montre le résultat de l'établissement de la condition 2 : les valuations sont projetées des contraintes vers les domaines en utilisant la procédure Project (mécanisme habituel de l'arc cohérence).

Sur les VCSP strictement monotones, la seule valuation α telle que $\alpha \oplus \alpha = \alpha$ est la valuation $\top$ qui ne peut être obtenue par une combinaison finie de coûts finis. Il est alors possible de donner une définition équivalente de la cohérence d'arc beaucoup pus simple, s'appuyant sur la notion de CSP sous-jacent [Coo02, CS02].

DÉFINITION 10 *Le* CSP sous-jacent *d'un VCSP* V *a les mêmes variables et domaines que* V *avec, pour chaque contrainte* $c_P \in C$*, une contrainte dure* c'_P *telle que* $\forall t \in \ell(P)(t \in c'_P \Leftrightarrow c_P(t) \prec \top)$ *(i.e., t n'est pas totalement interdit).*

On prouve alors [CS02] que :

THÉORÈME 1 *Si $\oplus$ est strictement monotone, un VCSP est arc cohérent ssi :*

1. *son CSP sous-jacent est arc cohérent.*
2. $\forall i \in X, \forall c_{ij} \in C, \forall a \in d_i$, *si* $c_i(a) \prec \top$ *alors* $\exists b \in d_j(c_{ij}(a,b) = \bot)$.

L'algorithme 2 permet alors d'établir la cohérence d'arc en temps $\theta(ed^2)$ si un algorithme optimal est utilisé pour établir la cohérence d'arc classique. La complexité spatiale est en $O(ed^2)$ aussi car la procédure Project modifie les valuations des paires des contraintes. Il est cependant simple de limiter cette complexité à $O(ed)$ en ne modifiant pas les valuations des paires de contraintes et en mémorisant dans $\Delta^-(i,a,c_{ij})$ la combinaison des valuations projetées de c_{ij} sur (i,a). La valuation effective de la contrainte c_{ij} sera calculée en ligne par $c_{ij}(a,b) \ominus (\Delta^-(i,a,c_{ij}) \oplus \Delta^-(j,b,c_{ij}))$. La procédure Project reste en $O(d)$. Il faut noter que cette structure de données à l'avantage supplémentaire d'éviter toute modification de la contrainte initiale qui peut donc parfaitement être définie en extension.

Algorithme 2: Établissement de la cohérence d'arc sur des CSP strictement monotones

```
Établir la cohérence d'arc dans le CSP sous-jacent;
pour chaque c_P ∈ C^+ faire
    pour chaque i ∈ P faire
        pour chaque a ∈ d_i faire Project(c_P, i, a);
```

Comme le montre [Sch00], le VCSP équivalent produit par l'algorithme n'est pas unique. Chaque fermeture définit ainsi un minorant associé $f_{\min}(V) = \bigoplus_{c_P \in C}[\min_t c_P(t)]$. Le problème d'existence d'une fermeture dont le minorant associé est maximum est malheureusement NP-difficile (cf. [CS02]).

3 Arc cohérence directionnelle

Afin de garantir la terminaison, l'arc cohérence sur les VCSP justes ne propagent que les valuations dites "absorbantes" ($\top$ dans le cas des structures strictement monotones). Une autre façon de garantir la terminaison consiste à imposer une direction de propagation pour chaque contrainte (comme cela était fait sur les *directed arc inconsistency counts* [Wal95]). Il devient alors possible de propager des valuations quelconques via l'utilisation conjointe de Project et Extend, sans perdre la terminaison. Ainsi, l'arc cohérence directionnelle peut être plus forte que la cohérence d'arc dans les VCSP.

Considérons le MAX-CSP de la figure 3(a). Il contient une contrainte binaire qui interdit la paire $((1,b)(2,b))$ et 2 contraintes unaires qui interdisant les valeurs $(1,a)$ et $(2,a)$. Ce VCSP est arc cohérent et le minorant associé $f_{\min}(V)$ vaut 0. Il est possible d'appliquer Extend à $(1,a)$ pour obtenir le VCSP équivalent 3(b) qui n'est plus arc cohérent. On peut alors appliquer Project sur $(2,b)$ et obtenir le VCSP 3(c) avec un minorant $f_{\min}(V) = 1$: par rapport à l'arc cohérence simple, l'arc cohérence directionnelle peut *déplacer* les poids entre variables. Cela peut permettre, comme l'a montré l'exemple ci-dessous, d'augmenter le minorant fourni par l'arc cohérence simple.

Sur la base de cette idée, on peut par exemple décider d'orienter les contraintes afin de faire circuler les valuations des contraintes unaires selon un graphe orienté sans circuit qui peut être défini sur la base d'un ordre des variables.

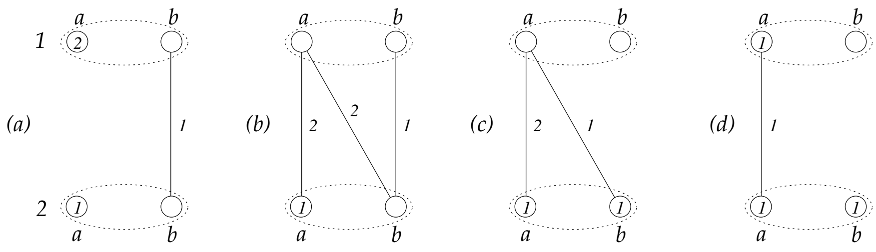


FIG. 3 – Établir l'arc cohérence directionnelle

DÉFINITION 11 *Un VCSP binaire est* directionnel arc cohérent *selon un ordre* $<$ *sur les variables si* $\forall c_{ij} \in C^+$ *telle que* $i < j, \forall a \in d_i$

$$c_i(a) = \min_{b \in d_j}(c_i(a) \oplus c_{ij}(a,b) \oplus c_j(b))$$

L'arc cohérence directionnelle peut être établie simplement sur tout VCSP juste au moyen de la procédure DAC de l'algorithme 3 (cf. [CS02]). Comme pour la cohérence d'arc, cette procédure est en temps $\theta(ed^2)$. La complexité spatiale peut encore une fois être ramenée à $O(ed)$ en mémorisant, en sus des Δ^- déjà présentés, des $\Delta^+(i,a,c_{ij})$, combinaisons des valuations étendues par Extend de (i,a) vers c_{ij}. La valuation effective de la contrainte c_{ij} sera calculée en ligne par $c_{ij}(a,b) \ominus (\Delta^-(i,a,c_{ij}) \oplus \Delta^-(j,b,c_{ij})) \oplus (\Delta^+(i,a,c_{ij}) \oplus \Delta^+(j,b,c_{ij}))$. La procédure Extend se simplifie alors, en $O(1)$. Encore une fois, cette structure de données permet de travailler sur des contraintes en extension, sans qu'il soit nécessaire de modifier les contraintes intiales. En pratique, il est possible de fusionner les structures Δ^- et Δ^+ en une unique structure Δ (la différence de Δ^+ et de Δ^-) et de simplifier ainsi les calculs.

Algorithme 3: Établir la cohérence d'arc directionnelle

Procedure DAC()
- **pour** $(i \leftarrow (n-1)$ à $1)$ **faire**
  - **pour chaque** $(c_{ij} \in C$ t.q. $j > i)$ **faire**
    - $\{$Début de propagation de $c_{ij}\}$;
    - **pour chaque** $(b \in d_j)$ **faire** Extend(j, b, c_{ij});
    - **pour chaque** $(a \in d_i)$ **faire** Project(c_{ij}, i, a);
    - $\{$Fin de propagation de $c_{ij}\}$;

L'arc cohérence directionnelle consiste donc à tenter de déplacer toutes les pénalités du problème vers une variable racine. Pour cela, partant d'une variable i de moins en moins profonde, on fait passer les poids disponibles au niveau unaire dans une variable plus profonde j dans la contrainte c_{ij} puis on projette ensuite cette contrainte sur i : toutes les pénalités de j et de c_{ij} récupérables sur j sont ainsi déplacées vers i. On répète ce mécanisme de $i = n-1$ jusqu'à la variable racine. Il est possible de voir l'arc cohérence directionelle comme une sophistication des *mini-buckets* introduits dans [Dec97]. En utilisant le même ordre des variables, et pour un paramètre de mini-bucket égal à 1, l'arc cohérence directionelle fournit un minorant identique à celui fourni par les mini-buckets. L'avantage de l'arc cohérence directionnelle réside dans le fait qu'elle ne fournit pas qu'un minorant mais aussi un problème équivalent simplifié sur lequel la recherche de solutions peut se poursuivre.

Un VCSP directionnel arc cohérent n'est pas nécessairement arc cohérent. C'est le cas du CSP de la figure 3(c). On montre [Coo02] qu'il est toujours possible d'établir simultanément la cohérence d'arc et la cohérence d'arc directionnelle. Appliquée au CSP de la figure 3(c), on obtient ainsi le CSP de la figure 3(d). On dit que le CSP est fortement directionnel arc cohérent. Sur les structures strictement monotones, cette propriété peut s'établir par l'algorithme suivant [Coo02] :

Algorithme 4: Établir la cohérence d'arc directionnelle complète dans les VCSP strictement monotones

Procedure FDAC()
- Établir la cohérence d'arc dans le CSP sous-jacent;
- **pour** ($j \leftarrow n$ à 2) **faire**
  - **pour chaque** ($c_{ij} \in C$ s.t. $i < j$) **faire**
    - **pour chaque** ($b \in d_j$) **faire** Project(c_{ij}, j, b);
  - **pour chaque** ($c_{ij} \in C$ s.t. $i < j$) **faire**
    - **pour chaque** ($b \in d_j$) **faire** Extend(j, b, c_{ij});
    - **pour chaque** ($a \in d_i$) **faire** Project(c_{ij}, i, a);
    - **pour chaque** ($b \in d_j$) **faire** Project(c_{ij}, j, b);

L'intérêt d'une telle procédure est de pouvoir encore augmenter le minorant associé au problème. En effet, dans l'algorithme précédent (arc cohérence directionnelle), il se peut qu'après avoir injecté les penalités unaires de j dans c_{ij} et $c_{i'j}$ puis projeté ces contraintes sur i, il est possible par projection de c_{ij} et $c_{i'j}$ sur j de récuperer des coûts unaires, qui peuvent éventuellement permettre d'augmenter le minorant associé au problème.

Comme le suggère [Coo02], cette procédure peut être affinée en tentant de faire circuler les pénalités sur des chemins les plus courts possibles entre chaque variable et la première variable. Pour cela, on ordonne les variables selon leur distance croissante à la variable 1 et on insère une ligne supplémentaire de propagation au préalable pour tenter de propager avant tout l'information sur ces chemin les plus courts.

Algorithme 5: FDAC2 : exploiter des chemins plus courts

Procedure FDAC 2
- Établir la cohérence d'arc dans le CSP sous-jacent;
- **pour** ($j \leftarrow n$ à 2) **faire**
  - **pour chaque** ($c_{ij} \in C$ s.t. $i < j$) **faire**
    - **pour chaque** ($b \in d_j$) **faire** Extend(j, b, c_{ij});
    - **pour chaque** ($a \in d_i$) **faire** Project(c_{ij}, i, a);
  - **pour chaque** ($c_{ij} \in C$ s.t. $i < j$) **faire**
    - **pour chaque** ($b \in d_j$) **faire** Project(c_{ij}, j, b);
  - **pour chaque** ($c_{ij} \in C$ s.t. $i < j$) **faire**
    - **pour chaque** ($b \in d_j$) **faire** Extend(j, b, c_{ij});
    - **pour chaque** ($a \in d_i$) **faire** Project(c_{ij}, i, a);
    - **pour chaque** ($b \in d_j$) **faire** Project(c_{ij}, j, b);

4 Comparaison

Nous ne rappellerons pas la définition du minorant induit par les *reversible DAC* et défini dans [LMS99]. [Sch00] a montré que la cohérence d'arc *pouvait* toujours fournir un minorant au moins égal à ce minorant pour des VCSP justes.

La cohérence d'arc directionnelle présente théoriquement deux avantages sur la cohérence d'arc : elle est résout complètement les VCSP structurés en arbre et elle satisfait une propriété d'optimalité locale appelée *irréductibilité d'arc* :

DÉFINITION 12 ([CS02]) *Un VCSP binaire V est* arc-irréductible *par rapport au minorant $f_{\min}$ si $\forall J \subseteq X, |J| = 2$, pour tout VCSP V' dérivé de V par une transformation préservant l'équivalence sur J, $f_{\min}(V) \geq f_{\min}(V')$.*

Autrement dit, aucune transformation locale à une contrainte ne permettra seule d'augmenter le minorant. [CS02] montre qu'un VCSP binaire juste qui est directionnel arc cohérent est arc-irréductible. Naturellement, cette propriété s'applique aussi à la cohérence d'arc directionnelle complète.

Afin de mieux évaluer en pratique le potentiel de chacun des minorants induits, nous les avons évalués en pratique sur des réseaux de contraintes de type MAX-CSP aléatoires ainsi que sur un sous-problème de l'instance 06 des problèmes d'affectation de fréquences présentés dans [CdL+99] et définis par le Centre d'Électronique de l'Armement (CELAR) que nous remercions ici.

4.1 Problème aléatoires

Dans cette première évaluation, nous avons comparé les minorants sur des CSP binaires surcontraints traités comme des MAX-CSP. Une famille de CSP binaires aléatoires est caractérisée par $\langle n, d, p_1, p_2 \rangle$ où n est le nombre de variables, d le nombre de valeurs par variable, p_1 la densité du graphe définie par le ratio du nombre de contraintes par rapport à un graphe complet et p_2 la dureté des contraintes, définie par le ratio entre le nombre de paires interdites et le nombre de paires possibles. Les variables contraintes et les paires interdites sont sélectionnées aléatoirement [Pro94]. Les classes suivantes ont été étudiées :

1. $\langle 10, 10, 1, p_2 \rangle$,
2. $\langle 15, 5, 1, p_2 \rangle$,
3. $\langle 15, 10, 50/105, p_2 \rangle$,
4. $\langle 20, 5, 100/190, p_2 \rangle$,
5. $\langle 25, 10, 37/300, p_2 \rangle$,
6. $\langle 40, 5, 55/780, p_2 \rangle$.

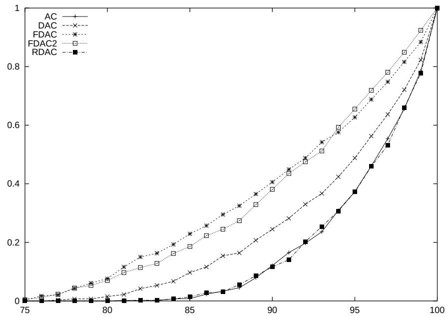
Notez que (1) et (2) correspondent à des graphes complets, (3) et (4) ont une densité moyenne et (5) et (6) sont peu denses. Pour chaque classe de problème et chaque valeur des paramètres, un échantillon de 50 problèmes a été généré.

Afin d'évaluer la force relative des minorants, chaque instance a été d'abord résolue à l'optimalité en utilisant l'algorithme PFC-MRDAC de [LMS99]. La valeur du minorant initial utilisé dans PFC-MRDAC, par l'établissement de la cohérence d'arc simple, de la cohérence d'arc directionnelle et de la cohérence d'arc directionnelle complète ont ensuite été évalués.

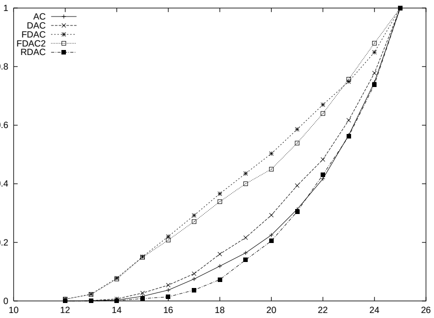
Dans le cas des cohérences d'arc directionnelles (complète ou non), nous avons tenter différents ordres de variables. Le fait que ces algorithmes de filtrage résolvent complètement les CSP en forme d'arbre lorsqu'un ordre topologique des variables est utilisé [CS02], nous ont conduit à tester un ordre des variables induit par une visite du graphe des contraintes en largeur d'abord, à partir d'un sommet de degré maximum. Cet ordonnancement s'effectue en temps linéaire dans la taille du graphe. Sur un graphe effectivement arborescent, on a ainsi une garantie d'optimalité. Les minorants correspondants sont notés DAC$_B$, FDAC$_B$ et FDAC2$_B$.

Comme proposé dans [Coo02], nous avons essayé un ordre plaçant les sommets dans un ordre de distance croissante à la première variable. Nous avons choisi comme première variable un centre du graphe (sommet qui minimise le maximum de la distance aux autres sommets). On a encore la garantie que l'ordre est un ordre topologique si le graphe est un arbre. L'obtention de l'ordre a été obtenue par l'algorithme de Floyd-Warshall (en $O(n^3)$). Les minorants correspondants sont notés DAC$_C$, FDAC$_C$ et FDAC2$_C$.

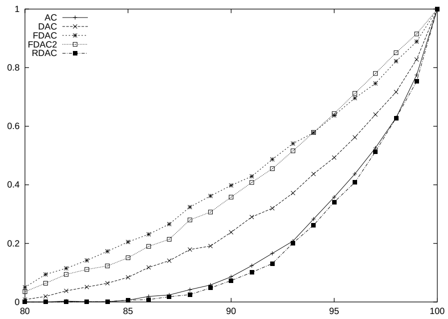
Au total, ce sont donc 12minorants qui ont été calculés sur chaque instance : AC, DAC, DAC$_B$, DAC$_C$, FDAC, FDAC$_B$, FDAC$_C$, FDAC2, FDAC2$_B$, FDAC2$_C$ et le minorant utilisé au démarrage de PFC-{M}RDAC [LMS99]. Les figures suivantes présentent la valeur moyenne du rapport entre la valeur du minorant et la valeur de l'optimum (toujours compris en 0 et 1 par définition).



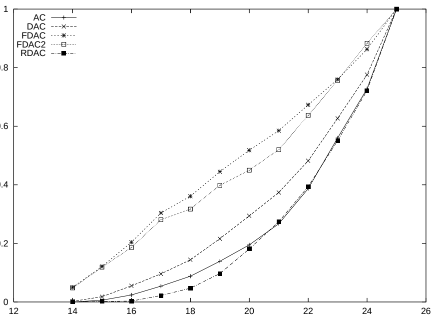
$\langle 10, 10, 1, p_2 \rangle$



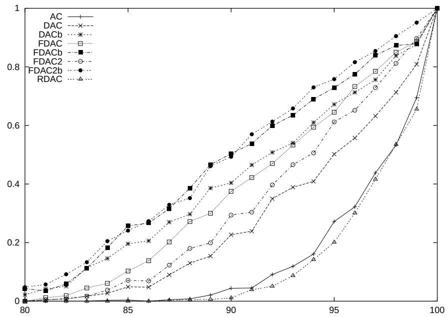
$\langle 15, 5, 1, p_2 \rangle$



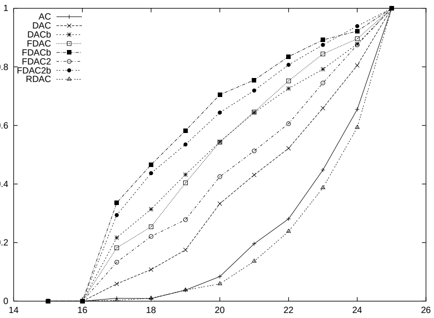
$\langle 15, 10, 50/105, p_2 \rangle$



$\langle 20, 5, 100/190, p_2 \rangle$



$\langle 25, 10, 37/300, p_2 \rangle$



. $\langle 40, 5, 55/780, p_2 \rangle$

Les résultats obtenus confirment l'intérêt pratique des versions directionnelles de la cohérence d'arc sur les MAX-CSP aléatoires qui fournissent toujours des minorants plus forts que l'arc cohérence simple. La question de la qualité d'un minorant résultant d'un compromis temps de calcul/force, il est aussi nécessaire de considérer ce critère. Les temps de calcul effectifs restent trop réduits sur des instances de cette taille (déjà importante pour l'algorithme de calcul de la valeur optimum). La simplicité des algorithmes utilisés donne cependant une idée assez précise de la complexité de chacun. Rappelons que lorsque les structures de données Δ^+ et Δ^- sont utilisées, Project est en $\theta(d)$ et Extend en $\theta(1)$. L'arc cohérence simple applique Project deux fois par contraintes. L'arc cohérence directionnelle applique Project et Extend une fois par contrainte. Enfin, la cohérence d'arc directionnelle complète applique trois fois Project et une fois Extend par contrainte.

Une des conclusion essentielle de cette étude est donc que l'arc cohérence directionnelle est apparemment un bon compromis temps/minorant. L'arc cohérence directionnelle complète permet de pousser le minorant un peu plus loin mais devrait être environ trois fois plus chère à établir. De façon générale, l'algorithme FDAC2, censé améliorer FDAC, n'apporte quelque chose que dès lors que les contraintes sont très dures. Il ne semble pas présenter un intérêt majeur vu le travail de propagation supplémentaire nécessaire.

L'arc cohérence simple est *a priori* dominée par l'arc cohérence directionnelle sur les deux points : complexité et valeur du minorant et semble donc moins intéressante. Elle domine tout de même sensiblement le minorant utilisé jusqu'ici en début de recherche de PFC-{M}RDAC [LMS99]. Si cette différence semble faible relativement aux gains induits par la cohérence d'arc directionnelle, il est vraisemblable que le cadre des MAX-CSP non pondérés (poids tous égaux à 1) est parmi les moins favorable pour mettre en évidence la supériorité de l'arc cohérence.

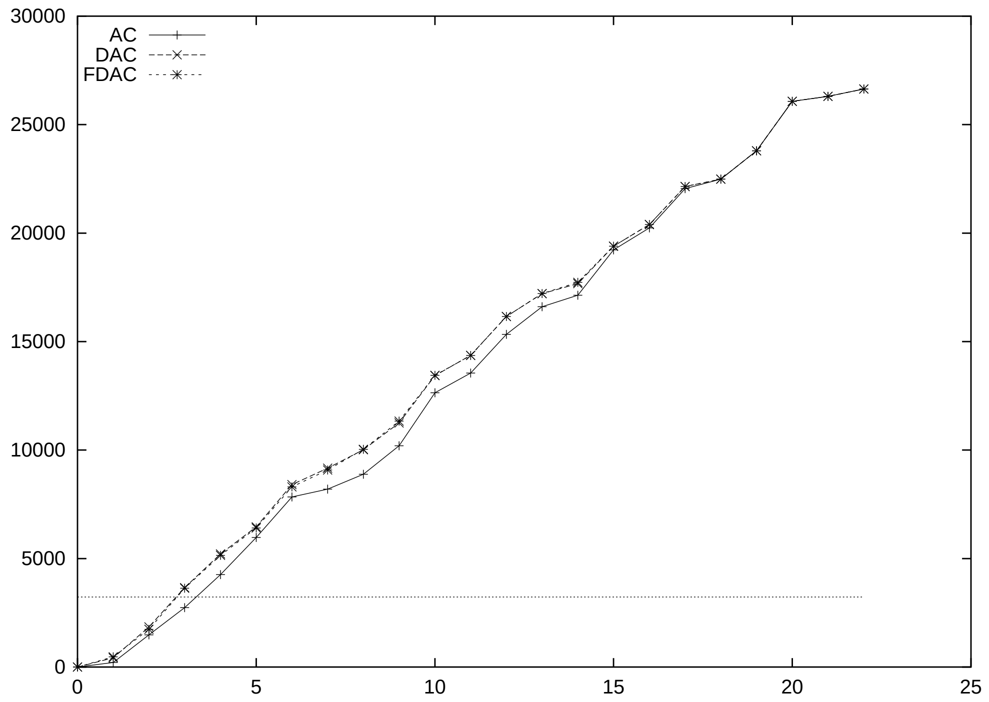
Sur les problèmes denses ou moyennement denses, la différence induite par les heuristiques de propagation (ordre en largeur, orientation vers le centre) ne produisent pas d'effets notables et sont donc omis des figures pour les rendre plus lisibles. Ces heuristiques très simples se montrent toutes les deux redoutablement efficaces sur les instances dont les graphes sont peu denses. Les deux heuristiques donnent des résultats très proches, nous n'avons, pour des raisons de clarté, presenté que les résultats obtenus avec l'heuristique d'ordonnancement en largeur d'abord. Ces heuristiques permettent à la cohérence d'arc directionnelle d'atteindre la qualité de la cohérence d'arc directionnelle complète (sans heuristique) et améliore tout autant l'arc cohérence directionnelle complète.

Ces résultats préliminaires demanderaient une étude plus complète (prise en compte d'autre critères : degré, valuations quand elles ne sont pas toutes égales à 1..., autres types de problèmes, aléatoires ou non).

4.2 Affectation de fréquence

Afin d'étudier aussi le comportement de ces minorants sur des problèmes non aléatoires, le sous-problème 4 du problème CELAR06 [CdL+99] a été utilisé. Les valeurs des minorants induits par l'arc cohérence simple, l'arc cohérence directionnelle et l'arc cohérence directionnelle complète sont toutes nulles sur ce problème. Pour mieux cerner l'efficacité de ces différents minorants dans le contexte d'une recherche arborescente, nous avons généré des problèmes de plus en plus contraints en affectant successivement chaque variable avec une valeur aléatoire tiré uniformément dans le domaine. Le problème ayant 22 variables, chaque série d'instanciation génère donc une série de 23 problèmes de plus en plus contraints. Au total, ce sont 10 000 séries de ce type qui ont été générées. L'ordre d'instanciation des variables choisi est un ordre statique (heuristique de largeur inverse, les ex-æco étant ordonnés par la largeur, cf. [LMS99]). L'ordre utilisé pour les versions directionnelles de la cohérence d'arc est l'ordre d'instanciation (mais ce n'est pas une obligation en pratique). Après chaque instanciation, la cohérence locale est établie et la valeur du minorant calculée. La moyenne du minorant à chaque niveau sur les 10 000 séries ainsi que la valeur de la solution optimum pour cette instance (3230) est indiquée sur le graphique.

Sur cette figure, aucune différence sensible n'a pu être observée entre FDAC et DAC. L'arc cohérence directionnelle semble dans ce cas un choix idéal. Dans le cadre d'une preuve d'optimalité, qui est souvent la partie la plus lourde de la recherche d'une solution optimale par un algorithme de type « séparation-évaluation », les différences observées entre l'arc cohérence simple, l'arc cohérence directionnelle et la valeur de la solution optimale laisse penser que l'utilisation de l'arc cohérence directionnelle pourrait permettre d'économiser un niveau de recherche (44 valeurs) dans l'arbre de preuve puisque le minorant moyen induit par DAC dépasse la valeur de l'optimum dès lors que 3 variables sont affectées alors que 4 variables sont nécessaires en moyenne dans le cas de



l'arc cohérence.

Aucune comparaison n'a été effectuée avec les minorants utilisés par PFC-$\{$M$\}$RDAC car ces minorants effectuent une propagation supplémentaire pendant l'exécution que n'effectuent pas (encore) les algorithmes que nous avons évalués : si le minorant associé à une valeur (i, a) d'une variable future (que l'on peut simplement définir par exemple par $f_{\min}^{ia} = c_{\varnothing}() \oplus \sum_{j \neq i}[\min_{b \in d_j}(c_j(b))] \oplus c_i(a))$ dépasse la valeur de la meilleure solution connue alors on peut effacer cette valeur. La définition de nouveaux algorithmes effectuant la propagation de ces effacements dans le cadre des propriétés de cohérence d'arc et de leurs versions directionnelles reste un travail à faire.

L'existence d'un majorant peut simplement être pris en compte dans les CSP valués en travaillant sur une structure de valuation telle que celle utilisée dans l'exemple de la figure 2 : une valuation finie est prise comme élément maximum $\top$. Dans ce cas, toute valeur dont le minorant associé atteint la valuation finie $\top$ sera effacée. Malheureusement, la structure de valuation n'est plus monotone et un algorithme général tel que l'algorithme d'arc cohérence généralisé proposé dans [CS02] doit être utilisé. Ce travail a été partiellement entamé dans [Lar02] dans le cas mes MAX-CSP pondérés..

Conclusion

L'intérêt pratique des versions directionnelles de la cohérence d'arc dans les CSP valués est clairement établi. La suite naturelle de ces travaux est de construire un algorithme de type « séparation-évaluation » basé sur ces minorants issus de la cohérence d'arc directionnelle. Il est clair qu'il reste encore un chemin non négligeable à parcourir avant d'atteindre ce but. Il faudra surtout définir des algorithmes dédiés au problème du maintien de ces propriétés suite à l'effacement de valeurs qui peuvent provenir d'origines plus variées que dans le cas des CSP classiques : instanciation, augmentation des coûts unaires jusqu'au niveau du majorant, augmentation du minorant, diminution du majorant. Néanmoins, il n'est pas abusif de penser ainsi aboutir à un algorithme aussi efficace mais plus élégant que l'algorithme PFC-MRDAC introduit dans [LMS99].

Remerciements : je remercie le relecteur qui par ses remarques critiques a peut-être permis de rendre ce papier un peu moins abscons qu'il n'était à l'origine. Il n'est pas toujours évident de s'en rendre compte. Je remercie aussi J. Larrosa pour m'avoir indiqué le lien fort qui existe entre arc cohérence directionnelle et mini-buckets d'ordre 1.

Références

[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.

[CdL+99] B. Cabon, S. de Givry, L. Lobjois, T. Schiex, and J.P. Warners. Radio link frequency assignment. *Constraints Journal*, 4 :79–89, 1999.

[Coo02] Martin C. Cooper. Reduction operations in fuzzy or valued constraint satisfaction. *Fuzzy Sets and Systems*, 2002. To appear.

[CS02] M. Cooper and T. Schiex. Arc consistency for soft constraints. *submitted to JACM*, 2002. see arXiv.org/abs/cs.AI/0111038.

[Dec97] Rina Dechter. Mini-buckets : A general scheme for generating approximations in automated reasoning. In *IJCAI*, pages 1297–1303, 1997.

[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.

[Kos99] Arie M.C.A Koster. *Frequency assignment : Models and Algorithms*. PhD thesis, University of Maastricht, The Netherlands, November 1999. Available at www.zib.de/koster/thesis.html.

[Lar02] J. Larrosa. On arc and node consistency in weighted csp. In *Proc. AAAI'02*, Edmondton, (CA), 2002.

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

[Pro94] Patrick Prosser. Binary constraint satisfaction problems : Some are harder than others. In *Proc. of the 11st ECAI*, Amsterdam, The Netherlands, 1994.

[Sch00] T. Schiex. Arc consistency for soft constraints. In *Principles and Practice of Constraint Programming - CP 2000*, volume 1894 of *LNCS*, pages 411–424, Singapore, September 2000.

[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.

[Wal95] 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, 1995.