NSCSP : DEFINITION ET RESOLUTION PAR TRANSFORMATION

Alexis Anglada,
Doctorant en programmation par contraintes et décision
alexis.anglada@dassault-aviation.fr , + 33 1 47 11 48 19

Adresses professionnelles
DASSAULT AVIATION ★ 78 Quai Marcel Dassault ★ 92552 Saint Cloud Cedex, France.
Laboratoire informatique Paris 6 (LIP6) ★ Rue du capitaine Scott ★75015 Paris, France.

Résumé : L'expressivité des problèmes de satisfaction de contraintes discrets (CSP) a été augmentée ces dernières années par l'introduction de la gestion de l'incertitude et des préférences (cf. les cadres SCSP(CSP à base de demi-anneau) et VCSP (CSP valué)) permettant ainsi d'appréhender de manière plus flexible les problèmes réels. Cependant ces derniers comportent souvent des composantes continues. Pour prendre en compte cette spécificité, la communauté développe activement la théorie des CSPs continus (Numerical CSP noté NCSP). Dans ce poster

nous définissons une extension du paradigme des problèmes de satisfaction de contraintes flexibles aux domaines continus dans le cadre des demi-anneaux. Nous montrons comment résoudre les NCSPs flexibles ainsi définis par transformation en un NCSPs classiques.

Summary : Recent extension of the Constraint Satisfaction Problem (CSP) paradigm are the soft CSP framework and the numerical CSP (NCSP) framework. The first one addresses the management of data uncertainty and the expression of preferences. The second one addresses numerical problem solving. In order to address real problems we wish to combine these two paradigm. Therefore we will define in this poster an extension of the soft CSP paradigm to the continuous domains within the semirings' framework. We will show how to solve soft NCSPs by transforming them into NCSPs.

Mots clés : Intelligence artificielle , Programmation par contraintes, Flexibilité, Problème de satisfaction de contraintes numériques.

Key words : Artificial intelligence, Constraint Programming, Softness, Numerical constraint satisfaction problem.

NSCSP : Définition et résolution par transformation

1 - INTRODUCTION

Les problèmes de satisfaction de contraintes (CSP) permettent de modéliser de nombreux problèmes réels tel que le dimensionnement ou l'allocation de ressources. Or, dans les cas réels, on retrouve souvent des données incomplètes, des spécifications imprécises, des problèmes sur-contraints. Ces situations ne sont pas prises en compte par les CSPs classiques, aussi a-t-on cherché à introduire de la flexibilité, via les contraintes molles ou flexibles, dans la programmation par contraintes. Les travaux en domaine fini ont conduit à dégager deux cadres : les CSPs valués (VCSP [SFV97]) et les CSPs à base de demi-anneau (Semi-ring based CSP (SCSP) [BMR95]). Il a été démontré que tout VCSP peut s'exprimer par un SCSP dans [BMR+99]. Bien qu'ils soient en adéquation avec la modélisation de problèmes réels, les CSPs continus (NCSP pour Numerical CSP) n'ont pas bénéficié de cette évolution. Aussi nous montrerons dans un premier temps comment étendre les SCSPs au cas continu en généralisant cette théorie. Cette extension permettra d'exploiter toute l'expressivité de la combinaison des SCSPs et des domaines continus, nous noterons le cadre ainsi obtenu NSCSP pour numerical semi-ring based CSP. Comme l'ont montré les auteurs de [BCR02] et de [BGR00], il est souvent difficile de résoudre un SCSP ; pour faciliter la résolution des NSCSPs, nous proposerons, dans un deuxième temps, une transformation de ces derniers en NCSPs classiques. Et nous montrerons que les solutions du NCSP obtenu sont les mêmes que celles du NSCSP d'origine.

Ce papier s'organise comme suit : dans la partie 2, après avoir rappelé quelques notations et définitions, nous définirons les NSCSPs. Dans la section 3, nous définirons notre méthode de transformation et prouverons la validité des solutions obtenues. Avant de conclure, nous parlerons de notre implémentation dans la section 4 et nous y montrerons en exemple résolu de NSCSP.

2 - VERS LES NSCSPS

Après un bref rappel des définitions utiles au cadre des SCSPs, nous donnerons une définition des contraintes flexibles continus et définirons ce qu'est un CSP continu à base de c-demi-anneau (NSCSP).

2.1 - Rappels

Dans la suite de ce document nous noterons, X l'ensemble des variables d'un problème, D l'ensemble des domaines associés à celles-ci, C l'ensemble des contraintes. $x_i \in X$ est une variable, d_{xi} est son domaine et c_i est une contrainte. Les SCSPs et les NSCSPs repose sur l'utilisation d'une structure mathématique : le c-demi-anneau.

Définition 1: *Un c-demi-anneau est un tuple $(A,\oplus,\otimes,0,1)$ tel que :*

A est un ensemble et $0 \in A$, $1 \in A$

$\oplus$, appelée opération additive, est commutative (i.e : $a \oplus b = b \oplus a$), associative ($a \oplus (b \oplus c) = (a \oplus b) \oplus c$), idempotent ($a \oplus a = a$), 0 est son élément neutre ($a \oplus 0 = a = 0 \oplus a$) et 1 son élément absorbant($a \oplus 1 = 1 = 1 \oplus a$)

$\oplus$, appelée opération multiplicative, est commutative, associative , 1 est son élément neutre et 0 son élément absorbant.

$\oplus$ est distributive par rapport à $\oplus$ (i.e : $\forall a,b,c \in A$, $a \oplus (b \oplus c) = (a \oplus b) \oplus (a \oplus c)$)

Un c-demi-anneau est muni d'un ordre partiel ($\leq_A$) et ses opérations sont monotones sur cet ordre.

Pour définir un SCSP, il est nécessaire de garder à l'esprit qu'une contrainte flexible est une relation qui, à chaque tuple de valeurs des variables sur lesquelles elle porte, associe une valeur dans l'ensemble A, support du c-demi-anneau, cette définition fut introduite dans [BMR95].

Définition 2: *Un SCSP est un tuple (X,D,C,A) avec (X,D,C) un CSP où les contraintes ont valeur dans A, un c-demi-anneau.*

2.2 - NSCSP

Les NSCSPS sont , à l'instar des NCSPs pour les CSPs, une extension des SCSPs. Comme pour leur homologues classiques, les contraintes exprimées au sein d'un NSCSP le sont en extension, car les domaines des variables sont des intervalles et possèdent donc une infinité de valeur. Aussi définissons nous les contraintes à l'aide d'une fonction et non comme une relation. Cette définition des contraintes se rapproche de celle fonctionnelle vue dans [BMR02].

Définition 3: *Une contrainte est un couple (def,con) où :*

- $con \in X$
- *def est une fonction*

$$def : \prod_{i \mid x_i \in con} d_{xi} \rightarrow A$$

où A est un ensemble de valeurs (ex : {vrai,faux}, [0,1], $\mathbf{R}$ *),* $\prod d_{xi}$ *le produit cartésien des domaines des variables sur lesquelles porte la contrainte. def est appelée fonction de satisfaction de la contrainte.*

Un exemple de contrainte flexible :

$$con = \{x,y\} \text{ , } d_x = d_y = [0,10]$$

$$[1,10] \rightarrow [0,1]$$

$$def : x, y \rightarrow \begin{cases} 1 & si \quad x < y \\ \dfrac{1 - |x - y|}{10} & autrement \end{cases}$$

On remarque que les définitions des contraintes en extension pour les domaines finis sont un cas particulier de cette définition. La définition d'un SCSP se généralise donc ainsi :

Définition 4: *Numerical Semi-ring based CSP*

Un NSCSP est un tuple P=(X, D, C, A) où (X, D, C) est un NCSP et A un c-demi-anneau. C est l'ensemble des contraintes définies selon le couple (def, con) de la définition 3.

Définition 5: *Solutions d'un NSCSP*

Une solution d'un NSCSP est une instanciation globale de X associée à sa valeur dans A. Cette valeur s'obtient par combinaison des valeurs de satisfaction de toutes les contraintes du problème. Cette combinaison s'obtient par application de l'opération multiplicative au résultats de toutes les résultats des foncions de satisfaction.

Une solution optimale d'un NSCSP est une solution qui a la "meilleure" satisfaction au sens de $\leq_A$.

Les instances des SCSPs montrés dans [CR00] s'étendent aux NSCSPs en gardant les mêmes c-demi-anneaux si les contraintes sont exprimées avec des fonctions de satisfaction comme prévu dans la définition 3.

3 - RESOLUTION PAR TRANSFORMATION

Les CSPs ainsi que les NCSPs sont résolu à l'aide d'algorithme de consistance. Pour les SCSPs, les auteurs de [BGR00] ont explicité les propriétés nécessaires de l'opération multiplicative du demi-anneau pour assurer la terminaison des algorithmes de consistance locale. Dans [BCR02] les auteurs définissent une autre méthode de résolution des SCSPs: l'abstraction. L'idée est d'exprimer un SCSP par un ensemble de CSP et de résoudre ces CSPs de manière itérative. Il existe des solveurs efficace de NCSP et nous voulons nous servir de cet avantage pour résoudre les NSCSPs. Nous proposons donc une technique de transformation inspirée de ce travail d'abstraction et des technique de réification utilisé dans le cadre des CSPs hiérarchiques. Dans un premier temps nous expliciterons notre méthode de transformation des NSCPs en NCSPs. Et dans un deuxième temps, nous montrerons l'équivalence entre les solutions du NSCSP et celles du NCSP qui en est issu.

3.1 Définition

Pour transformer un NSCSP en NCSP, il faut exprimer les contraintes flexibles via des contraintes dures. Le cas évident est celui où A le c-demi-anneau est ({vrai,faux},$\vee$, $\wedge$,faux,vrai) car toutes les contraintes sont déjà des contraintes dures, en effet ce c-demin-anneau est celui de (N)CSPs classiques. Pour la suite de cet article nous supposerons que le c-demi-anneau est soit ({vrai,faux},$\vee$, $\wedge$,faux,vrai) soit que $A \subseteq \mathbf{R}$; ainsi nous serons sûrs de pouvoir effectuer notre transformation car A est un ensemble exprimable par un intervalle comme prévu dans un NCSP. De plus nous posons comme notation : con_j l'ensemble des variables sur lesquelles porte la

contrainte c_j , def_j la fonction de définition de cette contrainte.

Nous supposons que nous avons une expression analytique pour chaque fonction de satisfaction des contraintes du problème.

Il existe deux cas de transformation d'une contrainte flexible selon sa fonction de satisfaction:

- la fonction de satisfaction à une expression analytique unique
- la fonction de satisfaction est "définie par morceaux"

Commençons par le cas de transformation le plus simple : une expression analytique unique. Pour obtenir une contrainte dure, il suffit d'introduire une variable supplémentaire, noté z_j , de domaine A, l'ensemble du c-demi-anneau, et dont la valeur sera donnée par la fonction de satisfaction selon l'expression :

$$z_j = def_j\ (con_j\ )$$

Prenons, dans le cadre des CSPs flous ou fuzzy CSPs (FCSP) A = [0,1], la contrainte flexible suivante :

$$c_1 : con_1 = \{ x,y \}\ ,dx = [1,10] = dy$$

$$def_1 : [1,10]^2 \rightarrow [0,1]$$

$$x,y \rightarrow \frac{1-|x-y|}{10}$$

Elle s'exprime par la contrainte dure suivante :

$$z_1 = \frac{1-|x-y|}{10}$$

$$\text{avec} \begin{cases} d_{z_1} = [0,1] \\ d_x = [1,10] \\ d_y = [1,10] \end{cases}$$

Notons qu'ici notre hypothèse $A \subseteq R$ permet bien d'avoir un domaine continu pour z_1 représentable par un intervalle.

La transformation se complique pour le cas où la fonction de satisfaction est "définie par morceaux". Il est nécessaire d'avoir un mécanisme de contrainte conditionnelle. Nous exprimerons une contrainte conditionnelle par l'expression régulière suivante : H ~> C où H est une condition s'évaluant par vrai ou faux et C une contrainte dure. H ~> C est une contrainte conditionnelle au sens où les opérateurs de réduction des domaines associés à C ne sont utilisés lors de la résolution que si H est vrai. H est toujours évalué et aucun opérateur de réduction n'y est associé. Avec ce mécanisme, une fonction de satisfaction "par morceaux " est transformée en autant de contraintes conditionnelles qu'il y a de "morceaux " dans sa définition.

Exemple, toujours avec un cadre FCSP :

$$c_2 : con_2 = \{ x,y \}\ ,dx = [1,10] = dy$$

$$def_2 : [1,10]^2 \rightarrow [0,1]$$

$$x,y \rightarrow \begin{cases} 1 & si \quad x \le y \\ \dfrac{1-|x-y|}{10} & si \quad x > y \end{cases}$$

s'exprime par :

$$x \le y \quad \sim> \quad z_2 = 1$$

$$x > y \quad \sim> \quad z_2 = \frac{1-|x-y|}{10}$$

$$avec \quad d_{z_2} = [0,1] \quad dx = dy = [1,10]$$

On a ainsi défini la transformation d'une contrainte flexible de la définition 3 en contrainte dure. Mais cela ne suffit pas pour avoir un NCSP équivalent au NSCSP de départ. Pour finir, il faut ajouter une contrainte qui fera l'agrégation des valeurs de satisfaction des contraintes flexibles. Cette contrainte introduit une nouvelle variable de domaine A qui représente la valeur de satisfaction d'une solution. Cette contrainte s'exprime ainsi :

$$sat = \bigotimes_{j=1}^{m} z_i$$

où $\otimes$ est l'opération multiplicative du demi-anneau A du SCSP considéré, m le nombre de contraintes flexibles transformées et zi les variables de domaine A introduites par cette même transformation. Dans le cadre des NSCSPs, comme dans celui des SCSPs, les solutions souhaitées sont souvent les solutions optimales. Pour les obtenir grâce à notre transformation, il est nécessaire d'optimiser la valeur de la variable *sat*.

3.2 Équivalence des solutions

Pouvoir transformer et ainsi résoudre un NSCSP n'est efficace que si les solutions trouvées après transformation sont bien celles définit par le NSCSP transformé.

Proposition 6 *(équivalence des solutions)*

Les solutions d'un NSCSP et celles du NCSP obtenu par la transformation définie dans la section 3.1 sont équivalentes.

Preuve :

Toutes instanciation globales des variables du NSCSP est un tuple T =(t1, . . . ,tn) associé à une valeur de satisfaction k ∈ A défini par $k = \bigotimes_{j=1}^{m} def_j(con_j)$ dans le SCSP.

Dans le NCSP obtenu par abstraction, toute instanciation globale est un tuple

$$T' = (t'_1, \ldots, t'_n, z_1, \ldots, z_m, sat)$$

Avec $sat = \bigotimes_{j=1}^{m} z_i$ et z_j obtenu par la transformation.

A toute instanciation dans le NSCSP correspond une seule instanciation dans le NCSP abstrait, car les fonctions def_j , qui définissent les contraintes, donne une seule valeur pour chaque tuple de valeurs de con_j donc les valeurs t_i des variables définissent de façon unique les valeurs z_j des variables de satisfaction.

D'où pour tout tuple T, on a un tuple T' avec $t_i = t'_i$ et k = sat par définition de sat et k. On a ainsi une bijection entre les tuples du NSCSP et ceux du NCSP obtenu. D'où toute solution du NSCSP existe dans le NCSP obtenu par transformation et sa valeur de satisfaction est celle attribuée à la variable sat. De plus, si k est optimal alors l'image de la solution dans le NCSP est optimale et si une solution est optimale dans le NCSP, alors sat est optimale et k est optimal, donc la solution est optimale dans le NSCSP. Nous avons donc l'équivalence des solutions du NSCSP avec celle du NCSP obtenu par transformation ainsi que pour les solutions optimales.

4 IMPLÉMENTATION ET EXEMPLE

4.1 Implémentation

L'équipe Contraintes et Décision de DASSAULT-AVIATION développe un logiciel basé sur un solveur de NCSP: Constraint Explorer(CE). CE a pour objectif la modélisation et la résolution des problèmes de conception. Cet outil est le logiciel utilisé et amélioré dans le projet RNTL CO2 (acronyme de COnception par COntraintes). Une description du projet CO2 et de CE est faite dans [Zim01].

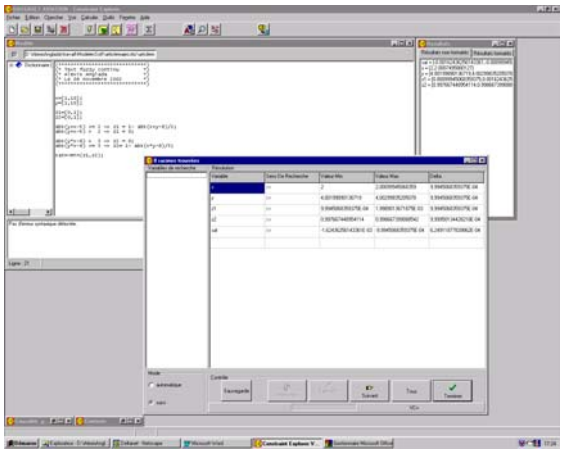

Figure 1 : Interface de Constraint Explorer

La figure 1 présente une vue de l'interface du logiciel de l'équipe Contrainte et Décision.

Ce logiciel fournit un mécanisme de contrainte conditionnelle à la fois dans son langage et dans son algorithme de résolution. La phase de propagation du solveur est assuré par un algorithme de Hull-consistance. Un algorithme de backtrack chronologique avec un ordre statique est à la base de la résolution. L'exploration de l'espace de recherche s'effectue par découpe dichotomique des domaines des variables. Le logiciel fournit aussi un mécanisme d'optimisation. Cette fonctionnalité est assurée par un algorithme de minimisation. Ce dernier fonctionne par réintroduction dans la recherche de la dernière solution trouvée comme borne supérieure de l'optimum. Ce schéma d'optimisation est dérivé de ceux expliqué dans [PM95] et de l'algorithme de branch and bound. Il a été adapté pour les domaines continus et implémenté dans CE comme expliqué dans [Let01].

4.2 Exemple

Nous allons maintenant présenter un exemple de NSCSP et certaines de ses solutions. Nous nous plaçons dans le cadre FCSP , le c-demi-anneau utilisé est ([0,1],max,min,0,1) .On a bien $[0,1] \subseteq R$, nous pouvons donc utiliser notre transformation. Notre problème est de trouver un couple de $x,y \in [1,10]^2$ tel que leur somme soit proche de 5 et leur produit de 8.

Les préférence sur les valeurs de x + y et x*y sont exprimé par les courbes de la figure 2.

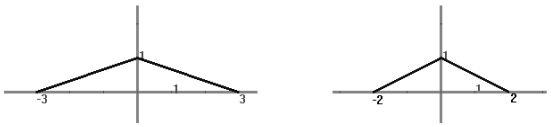

*Figure 2:Courbes de préférence. Les abscisses représentent les valeurs de x+y-5 à droite et x*y-8 à gauche.*

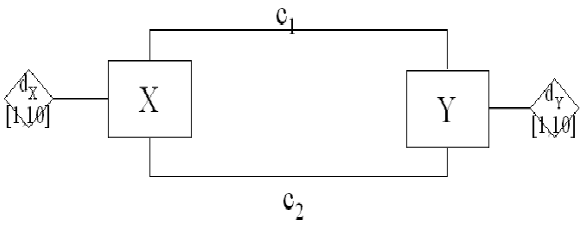

Figure 3:Exemple de NSCSP

Notre problème est schématisé par le NSCSP de la figure 3 avec c1 et c2 définies ci-dessous:

$$c_1 : con_1 = \{x, y\}$$

$$def_1 : [1,10]^2 \rightarrow [0,1]$$

$$x, y \rightarrow \begin{cases} 0 & si \quad |x+y-5| > 2 \\ \dfrac{1-|x+y-5|}{2} & si \quad |x+y-5| \le 2 \end{cases}$$

$$c_2 : con_2 = \{x, y\}$$

$$def_2 : [1,10]^2 \rightarrow [0,1]$$

$$x, y \rightarrow \begin{cases} 0 & si \quad |x*y-8| > 3 \\ \dfrac{1-|x*y-8|}{3} & si \quad |x*y-8| \le 3 \end{cases}$$

Les domaines de x et y sont d_x=d_y=[1,10].

Par transformation, nous obtenons le NCSP suivant :

$$d_x = d_y = [1,10], d_{z_1} = d_{z_2} = [0,1]$$

$$|x+y-5| \le 2 \sim> z_1 = \frac{1-|x+y-5|}{2}$$

$$|x+y-5| > 2 \sim> z_1 = 0$$

$$|x*y-8| \le 3 \sim> z_2 = \frac{1-|x*y-8|}{3}$$

$$|x*y-8| > 3 \sim> z_2 = 0$$

$$sat = \min(z_1, z_2)$$

La première ligne exprime le domaine de chaque variable.

Nous avons résolu ce NCSP grâce à notre implémentation en arrêtant la découpe dichotomique des domaines quand leur taille est de l'ordre de 1% de leur borne inférieur. La table 1 présente quelques exemples de résultats La figure 4 présentent une courbe 3D de la satisfaction globale d'une solution, en fonction des valeurs des variables x et y, issue des résultats obtenu grâce à notre implémentation. Nous avons ensuite cherché les solutions optimales qui correspondent au sommet de cette courbe. Leurs valeurs, obtenues avec le critère précédent de terminaison de découpe des domaines, sont présentées dans la table 2.

x	y	z_1	z_2	sat
[1,1.01]	[5.59,5.6]	[0.2,0.203]	[0.2,0.203]	[0.2,0.203]
[1.41,1.43]	[4.56,4.59]	[0.5,0.506]	[0.5,0.506]	[0.506,0.506]
[1.93,1.96]	[3.62,3.67]	[0.7,0.708]	[0.7,0.708]	[0.705,0.705]

Table 1:Exemples de solutions

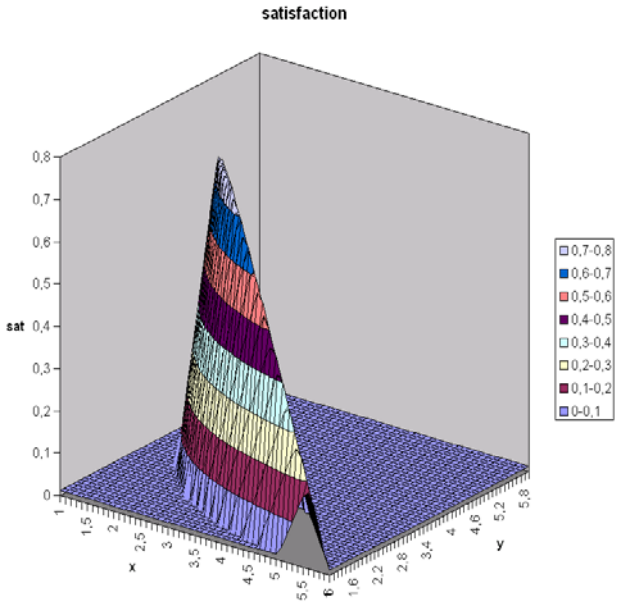

Figure 4:Courbe de satisfaction globale des solutions en fonction des valeurs de x et de y

x	y	z_1	z_2	sat
[2,6257; 2,6521]	[2,7744; 2,8023]	[0,7860; 0,7868]	[0,7860; 0,7874]	[0,7860; 0,7865]
[2,6522; 2,6789]	[2,7467; 2,7743]	[0,7860; 0,7872]	[0,7860; 0,7882]	[0,7860; 0,7869]
[2,6521; 2,6537]	[2,7743; 2,7759]	[0,7860; 0,7868]	[0,7860; 0,7874]	[0,7860; 0,7865]
[2,6790; 2,7058]	[2,7193; 2,7466]	[0,7860; 0,7875]	[0,7860; 0,7886]	[0,7860; 0,7870]
[2,6789; 2,6814]	[2,7466; 2,7491]	[0,7860; 0,7872]	[0,7860; 0,7883]	[0,7860; 0,7869]
[2,7058; 2,7331]	[2,6922; 2,7194]	[0,7860; 0,7875]	[0,7860; 0,7886]	[0,7860; 0,7870]
[2,7058; 2,7086]	[2,7194; 2,7222]	[0,7860; 0,7874]	[0,7860; 0,7886]	[0,7860; 0,7869]
[2,7331; 2,7606]	[2,6654; 2,6922]	[0,7860; 0,7874]	[0,7860; 0,7885]	[0,7860; 0,7869]
[2,7331; 2,7358]	[2,6922; 2,6949]	[0,7860; 0,7874]	[0,7860; 0,7885]	[0,7860; 0,7869]
[2,7606; 2,7884]	[2,6388; 2,6654]	[0,7860; 0,7871]	[0,7860; 0,7878]	[0,7860; 0,7867]
[2,7606; 2,7626]	[2,6654; 2,6674]	[0,7860; 0,7870]	[0,7860; 0,7879]	[0,7860; 0,7867]
[2,7884; 2,8023]	[2,6257; 2,6396]	[0,7860; 0,7864]	[0,7860; 0,7868]	[0,7860; 0,7863]

Table 2:Solutions optimales

5 - CONCLUSION ET TRAVAUX FUTURS

Nous avons étendu le paradigme des SCSPs au domaine continu et défini une résolution par transformation. Nous pouvons dorénavant utiliser la puissance d'expression de toutes les instances des SCSPs définies, car les c-demi-anneaux restent les mêmes. Aussi pensons nous développer un ou plusieurs c-demi-anneaux spécifiques au domaine continu. L'incertitude apporte des arguments qualitatifs aux solutions, nous pensons appliquer ces théories à la décision. Dans le domaine de la décision, l'approche multicritère joue un rôle important pour être au plus près des problèmes réels. Il serait intéressant de trouver un c-demi-anneau de dimension N pour traiter le multicritère ou d'avoir un c-demi-anneau permettant une agrégation "intelligente" des critères en un seul. Ces voies sont en cours d'exploration. D'autres peuvent être entreprises dans la théorie élargie pour les deux contextes discret et continu, voire sur des SCSPs mixtes

BIBLIOGRAPHIE

[BGR02] Stefano Bistarelli, Philippe Codognet, and Francesca Rossi. (2002) "Abstracting soft constraints: Framework, properties, examples." *Artificial Intelligence*, Vol. 139, n° 2.

[BGR00] Stefano Bistarelli, Rosella Gennari, and Francesca Rossi.(2000) "Constraint propagation for soft constraints: Generalization and termination conditions." *Principles and Practice of Constraint Programming*, p. 83–97.

[BMR95] S. Bistarelli, U. Montanari, and F. Rossi. (1995) "Constraint solving over semirings." *IJCAI-95*, p. 624–630.

[BMR+99] S. Bistarelli, U. Montanari, F . Rossi, T. Schiex, G. (1999) Verfaillie, and H. Fargier. "Semiring-based csps and valued csps : Framework, properties and comparaison. ", *Constraints*, Vol. 4, n° 3, p. 199–240.

[BMR02] S. Bistarelli, U. Montanari, and F. Rossi. (2002) "Soft concurrent constraint programming." *European Symposium on Programming*, p. 53–67.

[CR00] Philippe Codognet and Francesca Rossi. (2000) "Tutorial notes : Solving and programming with soft constraints : Theory and pra-tice." *ECAI2000*.

[SFV97] Thomas Schiex, Hélène Fargier, and Gérard Verfaillie. (1997),"Problèmes de satisfaction de contraintes valués. ",*Revue d'intelligence artificielle*, Vol. 11,n° 2, p. 339–373.

[PM95] Steven Prestwich and Shyan Mudambi.(1995), "Improved branch and bound in constraint logic programming." *Principles and Practice of Constraint Programming CP'95*, First International Conference, Cassis, France, September 19-22, p. 533–548.

[Let01] Grégory Letribot (2001), "Optimisation et études paramétriques dans le cadre de CSP numériques", *Rapport de stage de DESS d'Intelligence Artificielle de l'Université Paris VI*, Dassault-Aviation.