NSCSP : Definition and resolution by transformation

Alexis Anglada[1,2], Philippe Codognet[2], and Laurent Zimmer[1]

[1] Dassault Aviation,DGT/DPR/ESA,78, quai Marcel Dassault, 92552 Saint-Cloud Cedex, France
[2] Laboratoire informatique Paris 6 (LIP6), Rue du capitaine Scott, 75015 Paris, France.

Abstract. 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 paper 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. We will also describe how to define flexible constraints via graphical representations. Finally we will develop a multicriteria approach in this soft NCSPs framework. And we will conclude by an application on a relevant example.

1 Introduction

Recent extension of the Constraint Satisfaction Problems (CSP) paradigm are on the first hand Soft CSPs and on the other hand NCSPs. The first extension addresses the management of data uncertainty and the expression of preferences. Two theoretical framework have been proposed: Semiring based CSPs (SCSP in [BMR95]) and Valued CSPs (VCSP [SVF95]). The authors of [BMR$^+$99] have proved that any VCSP can be expressed by a SCSP. The second extension was introduced to handle continuous domain and numerical solving. Real problems often requires to simultaneously express uncertainty, preferences and to handle continuous domain and numerical constraint. Therefore we wish to merge this two extensions of the CSP paradigm. We will show how to extend SCSPs to the continuous case by generalizing this theory. This will allow us to use the whole expressivity of the combination of SCSPs and of the continuous domains. We call this framework NSCSP, for Numerical Semiring based CSP. As shown by the authors of [BCR02], [BGR00], it is often difficult to solve a SCSP. So we propose a transformation of NSCSPs in NCSPs. We demonstrate that the solutions of the resulting NCSP are the same as those of the NSCSP. The soft CSPs formalism makes it possible to use preferences. But it is often difficult to express preferences in an analytical form. We propose a graphical approach based on simple mathematical tools in order to define these preferences.

In a real problem, decision-making is rarely based on a single criterion. So we propose a way to use multicriteria optimization with constraint programming to address decision-making.
This paper is organized as follows: in part 2 some notations and definitions are given and we define NSCSPs. In the section 3, we define our transformation method and prove the validity of the resulting solutions. In section 4, we show the utility of using mathematical tools for a graphical representation of preferences. We explain how to take into account multicriteria objectives within NSCSPs. Before concluding, section 5 present our implementation and some result on a NSCSP.

2 From SCSP to NSCSP

2.1 Semiring and SCSP

In the rest of the paper, we use the following notation : X for the set of variables, D the set of domains, C the set of constraints, x_i will denote a variable, d_{x_i} its domain and c_j a constraint. Semiring-based CSPs extend CSPs with the ability to associate to each tuple or constraint a value in a domain with a semiring structure.

Definition 1. *A semiring S is a tuple $(A, +, \times, \mathbf{0}, \mathbf{1})$ such that :*

- *A is a set and $\mathbf{0}, \mathbf{1} \in A$*
- *$+$, called the additive operation, is a closed, commutative and associative operation such that $a + \mathbf{0} = a = \mathbf{0} + a$*
- *$\times$, called the multiplicative operation, is a closed and associative operation such that $\mathbf{1}$ is its unit element and $\mathbf{0}$ its absorbing element*
- *$\times$ distributes over $+$*

A c-semiring is a semiring such that $+$ is idempotent, $\times$ is commutative, and $\mathbf{1}$ is the absorbing element of $+$.

A c-semiring implicitely defines a partial order($\leq_A$) over the set A and its operations are monotone over it. We call SCSP a constraint problem over a constraint system(CS) as it was defined in [BMR95].

Definition 2. *A constraint system is a tuple $CS = (S, D, X)$, where S is a c-semiring, D is a finite set, and X is an ordered set of variables. Given a constraint system $CS = (S, D, X)$, where $S = (A, +, \times, \mathbf{0}, \mathbf{1})$, a constraint over CS is a pair (def,con), where $con \subseteq X$ and is called the type of the constraint, and "def": $D^k \rightarrow A$ (where k is the size of con, that is , the number of variables in it), and it is called the value of the constraint. Moreover, a constraint problem P over CS is a pair P=(C,con) where C is a set of constraint over CS and con $\subseteq X$.*

2.2 NSCSP

NSCSPs extends SCSPs like NCSPs does for CSPs. The prevoius definition of SCSP requires an extensive defintion for "def". Conversely in NSCSPs we need to express them intentionally because domains are intervals. We propose a functional definition for the constraints inspired by [BMR02].

Definition 3. *A constraint is a pair (def,con) where :*

- $con \subseteq X$
- def *is a function*

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

where A is a set of values (e.g. : $\{true, false\}$, $[0,1]$, $\mathbb{R}$*),* $\prod_{x_i \subset con} d_{x_i}$ *the cartesian product of the domain associated with the variables involved in the constraint. We call* def *the satisfaction function of the constraint.*

An example of soft constraint:

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

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

$$x, y \rightarrow \begin{cases} 1 & \text{if } x < y \\ 1 - \frac{|x-y|}{10} & \text{in other cases} \end{cases}$$

The extensive definition of constraint on discrete domains is a particular case of our definition. We generalise the notion of SCSP in this way :

Definition 4. *Numerical Semiring-based CSP*
A NSCSP is a tuple P=(X, D, C, S) where X is a set of variables, D a set of associated domains, C a set of constraints (defined as above) and S is a c-semiring $(A, +, \times, \mathbf{0}, \mathbf{1})$.

Definition 5. *Solutions*
For a NSCSP (X, D, C, S), a solution is a tuple of values of D associated with a value of A. This value is provide by the combination of the satisfaction values of all the constraints in C. Combination of constraints results from the application of the multiplicative operation to the results of all satisfaction functions.
An optimal solution is a solution who have the "best" value in A according to $\leq_A$.

3 Constraint Solving by transformation

CSPs are solved by consistency algorithms and NCSP too. For SCSP, [BGR00] gives properties for the multiplicative operation of the c-semiring to ensure the termination of the local consistency algorithm. In [BCR02] the authors define another way to solve SCSP: abstraction. The idea is to express SCSP by many CSPs an solve these CSPs iteratively . Efficient solvers for NCSP exist and we would like to take advantage of these to solve NSCSP. So in this part, we propose a transformation technique inspired by this work (but quite different) and the technique of reification from the hierarchical CSPs framework. We will first define our method to transform NSCSP in NCSP and then prove the equivalence of solutions for the two problems.

3.1 Definition

To transform a NSCSP into a NCSP, we must express soft constraints by hard constraints. In NCSP all domains are intervals included in $\mathbb{IR}$. Our transformation introduce new variables with A as domain. We will suppose in the following that $A \subseteq \mathbb{IR}$. In this case, we can represent elements of A by intervals like usual in NCSP. From now on we will note con_j the set of variables involved in the constraint c_j and def_j its satisfaction function.

We suppose we have the analytical expression of the satisfaction function for all the soft constraints. This analytical expression can be either defined over a single interval and thus expressed by a single formula, or expressed by many formulas, one for each interval of definition.

We begin with the first case. To obtain a hard constraint, we just introduce an extra variable, z_j whose domain is A, the value set of the c-semiring. We then add the following contraint to the problem :

$$z_j = def_j(con_j)$$

The basic idea is thus simply to reify the semiring-based valuation as a numerical equality constraint.

An example in the framework of fuzzy CSP(FCSP), with $A = [0, 1]$, is given below, with the soft constraint :

$$c_1 : con_1 = \{x, y\}, d_x = [1, 10] = d_y$$

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

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

After transformation, it can be expressed by the following hard constraint:

$$z_1 = \frac{1 - |x - y|}{10} \text{ with } \begin{cases} d_{z_1} = [0, 1] \\ d_x = [1, 10] \\ d_y = [1, 10] \end{cases}$$

For performing he transformation in the second case, we use *conditional constraints*. A conditional constraint is written $H \sim> C$ where H is a condition (i.e. a boolean expression) and C a hard constraint. $H \sim> C$ is a conditional constraint in the sense that only C is associated with the reduction operator. The consistency algorithm use this operator only if H is true. With this mechanism, we transform a satisfaction function defined by many expressions to many conditional constraints: one for each expression of the soft constraint.
Let us consider the example below, again in the FCSP framework :

$$c_2 : con_2 = \{x, y\}, d_x = [1, 10] = d_y$$

$$def : [1, 10]^2 \rightarrow [0, 1]$$
$$x, y \rightarrow \begin{cases} 1 & \text{si } x \leq y \\ \frac{1-|x-y|}{10} & \text{si } x > y \end{cases}$$

by transformation, we express it by:

$$x \leq y \sim> z_2 = 1$$
$$x > y \sim> z_2 = \frac{1 - |x - y|}{10}$$
$$\text{avec } d_{z_2} = [0, 1], d_x = [1, 10], d_y = [1, 10].$$

Now we must add a constraint to perform the combination of values of the soft constraints. This constraint introduces a new variable whose domain is A, representing the satisfaction (semiring value) of a solution. This constraint is expressed as:

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

where $\otimes$ is the multiplicative operation of the c-semiring S of the NSCSP, m the number of soft constraint transformed and z_j the new variables introduced by our transformation.

In the NSCSPs framework, as in the SCSPs, we look for optimal solutions and thus have to maximize the value of the variable *sat*.

3.2 Solution Equivalence

This transformation provides a good way to solve NSCSPs only if the solutions of the NCSP obtained after transformation are equivalent to those of the original NSCSP. This is straightforward to establish.

Proposition 1. *(Equivalence of solutions)*
The solution of a NSCSP and those of the NCSP obtained by the transformation defined in section 3.1 are equivalents.

Proof. A solution for a NSCSP is a tuple $T = (t_1, \ldots, t_n)$ associated with a satisfaction value $k \in A$ define by $k = \overset{m}{\underset{j=1}{\otimes}} def_j(con_j)$ in the NSCSP.
The solution of the NCSP obtained by transformation is a tuple

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

Where

$$sat = \overset{m}{\underset{j=1}{\otimes}} z_j \text{ et } z_j \text{ obtained by the transformation.}$$

Only one tuple in the NCSP corresponds to a tuple in the NSCSP. The satisfaction functions def_j defining the constraints assign one value to each tuple for variables of con_j. So the value t_i of variables define in aunique way the values z_j of the satisfaction variables.
For each tuple T we get a tuple T' with $t_i = t'_i$ and $k = sat$ by definition of sat and k. By this we establish a bijection between the tuples of the NSCSP and those of the NCSP obtained by transformation. So all solution of the NSCSP exist in the NCSP. The satisfaction value of the NCSP's solution is the one assigns to sat. Moreover if k is optimal then the image of the solution in the NCSP is optimal and if a solution is optimal in the NCSP then sat is optimal and k also, therefore the solution is optimal in the NSCSP. We get the equivalence between the NSCSP's solutions and those of the NCSP obtained by transformation, and the same follows for the optimal solutions. □

4 How to express preferences?

In a NSCSP the constraints are defined by their satisfaction functions def. These functions must be in analytical form. It is however hard to find the expression that corresponds to the preferences that the user ideally wants to express. There are many cases where we know the shape of the curve representing the desired preferences but we don't know their exact values. So we propose to use a graphical approach to define the satisfaction function. Consider the Fuzzy CSP framework. where preferences can be expressed by taking $\{[0, 1], max, min, 0, 1\}$ for the c-semiring of our NSCSP. We consider the soft constraint problem depicted by Figure 1. The goal is to find a pair of value for x and y such that their sum (c_1) is close to 5 and their product (c_2) close to 8. The notion of proximity doesn't provide directly an analytical expression. So we must define explicitly def_1 and def_2. For this we present two approaches: piecewise linear curves and Béziers curves.

4.1 Piecewise linear curves

This approach is inspired by the triangular and trapezoid forms used in fuzzy logic. The figure 2 presents an expression of the preference curves for our problem. It is easy to find the analytical formulas for this style of curves. We need

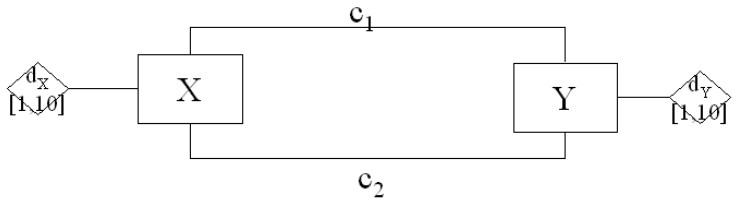

Fig. 1. Example of NSCSP with 2 variables (X,Y),their domain (d_X, d_Y) and 2 constraints (c_1, c_2)

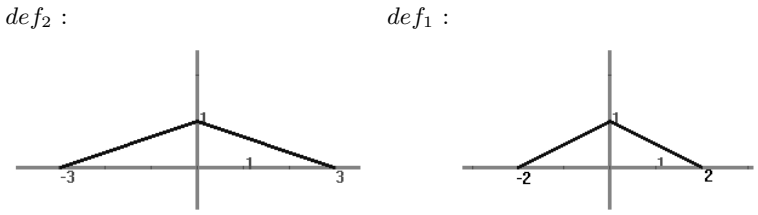

Fig. 2. Preferences' curves. X axis represents the value of $x * y - 8$ on the left side and the value of $x + y - 5$ on the right side. Y axis represents the satisfaction value of the constraints.

only two points by segment to determine the equation of each straight line support of each segment. With this method we find the satisfaction function for our constraint c_1 and c_2 :

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

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

$$x, y \to \begin{cases} 0 & \text{si } |x + y - 5| > 2 \\ \frac{1-|x+y-5|}{2} & \text{if } |x + y - 5| \leq 2 \end{cases}$$

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

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

$$x, y \to \begin{cases} 0 & \text{si } |x * y - 8| > 3 \\ \frac{1-|x-y|}{3} & \text{if } |x * y - 8| \leq 3 \end{cases}$$

The domains of x and y are $d_x = [1, 10] = d_y$.

When we have the analytical expressions, we can then apply the previously-defined transformation and obtain the following NCSP :

$$d_x = [1,10]; d_y = [1,10]; d_{z_1} = [0,1]; d_{z_2} = [0,1]; d_{sat} = [0,1];$$
$$|y + x - 5| \leq 2 \sim> z_1 = 1 - \frac{|x + y - 5|}{2};$$
$$|y + x - 5| > 2 \sim> z_1 = 0;$$
$$|y * x - 8| > 3 \sim> z_2 = 0;$$
$$|y * x - 8| \leq 3 \sim> z_2 = 1 - \frac{|y * x - 8|}{3};$$
$$sat = min(z_1, z_2);$$

The first line describes the domain of each variable. The graphical manipulation of the segment makes easier the definition of preferences : the user simply move, add or delete some points. However, this tool provides only curves of limited shapes.

4.2 Béziers curves

The Béziers curves are a particular case of piecewise polynomial functions. They can model many shapes depending on the number of control points used in their definition (see [Béz68]).The first and the last control points define respectively the beginning and the end of the curve. Intermediate points define the shape of curves by defining tangents. With only four control points we can already define many shapes as shown in the figure 3. Moreover this form are easily obtained by graphical manipulation (addition, moving and withdrawal of control point).

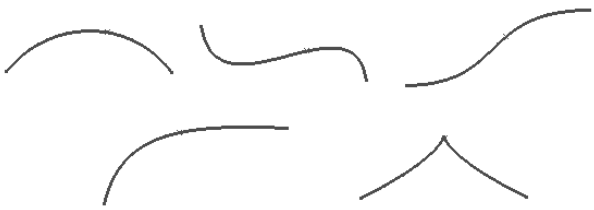

Fig. 3. Example of shape defined by Béziers' curves with four control points

This curves are obviously more expressive than piecewise linear functions. In particular they can capture a more accurate notion of proximity. For example, we see that the use of the first shape of figure 3 instead of those of the figure 2 avoid a stiff decrease of the preference value around the maximums. In return our transformation will be slightly more complicated, since the Béziers curves are defined by two functions: $X = f(t), Y = g(t)$ with $t \in [0,1]$ and $f(t), g(t)$

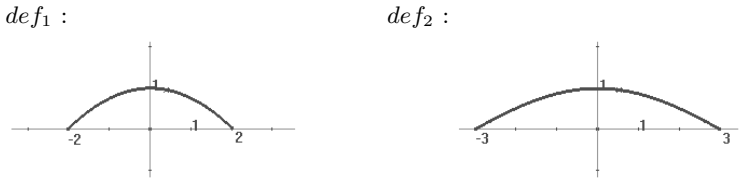

Fig. 4. Satisfaction's curves defined by Béziers' curves. X axis represents the value of $x * y - 8$ on the right side and the value of $x + y - 5$ on the left side. Y axis represents the satisfaction value of the constraints.

being polynomial expressions.

The right curve of Figure 4, is defined by :

$$\begin{aligned} t \in [0,1], X \in [-3,3] \\ X = 2 * t^3 - 3 * t^2 + 7 * t - 3 \\ Y = -4 * t^2 + 4 * t \end{aligned}$$

All these formulas must be introduced in the transformed NCSP in order to preserve equivalence with the original NSCSP. Let us consider $X = x * y - 8$ and $Y = z_1$, by introducing t_1 we obtain the following system to express the constraint:

$$\begin{aligned} t_1 \in [0,1] \\ x * y - 8 = 2 * t_1^3 - 3 * t_1^2 + 7 * t_1 - 3 \\ z_1 = -4 * t_1^2 + 4 * t_1 \end{aligned}$$

We do the same with the other constraint and we obtain the following NCSP.

$$\begin{aligned} t_1 \in [0,1], t_2 \in [0,1] \\ x * y - 8 = 2 * t_1^3 - 3 * t_1^2 + 7 * t_1 - 3 \\ z_1 = -4 * t_1^2 + 4 * t_1 \\ x + y - 5 = 4 * t_2^2 - 2 \\ z_2 = -4 * t_2^2 + 4 * t_2 \\ sat = min(z_1, z_2) \end{aligned}$$

Searching for the best solution of the original NSCSP now just amounts to solve this system and to optimize on the value of *sat*.
The simplicity for the graphical use of the Béziers curves and their different forms are appreciable to define the satisfaction functions for the NSCSPs.

4.3 Multicriteria Approach

Preferences, uncertainty allow a more flexible way to model real problem. But with a broad space of solution it remains difficult to choose which are the best. Most decision are based on multicriteria. In Decision community multicriteria optimization is used to select the best solutions among all solutions. We can see in the NSCSP framework the multicriteria approach as a specific case where the semiring is n-dimensional. N-dimensional semiring can be obtained by combining semiring, by example: Egalitarianism and Utilitarianism in [BMR95]. The question is always how to combine values of preferences and how to compare solutions in a multicriteria approach. To benefit from the results of the Decision community, we introduce the concept of predominance in the NSCSP paradigm.

Definition 6. *PreDominance*

$$\forall X, Y \in \mathbb{R}^n, X > Y \ ssi \begin{cases} \forall i = 1, \ldots, n \ X_i \geq Y_i \\ \exists i \in \{1, \ldots, n\} \ X_i > Y_i \end{cases}$$

In this definition $X > Y$ means X dominates Y.

An *efficient solution* is a solution which is dominated by no other solution. We said that such solutions are Pareto-optimal. The set of all efficient solutions is the Pareto-frontier. Several algorithms characterize the solutions in order to determine if they are efficient or not. In order to characterize solutions, they use mathematical formulas such as weighted sums. Starting from the set of all solutions, these functions (called generators) generate the Pareto set. Some of them require particular condition to be correct. In the constraint solving paradigm, the set of all solutions of a problem is intentionally defined by the constraint network. We wish to directly use these generators within the constraint solving algorithms. To test this approach, we have chosen in [Wie86] a generator which requires neither convexity assumption nor restriction on the solution space. The generator we chose is based on the Chebyshev norm which mathematical definition is (see also equation (34) in [Wie86]):

$$\forall x, y \in \mathbb{R}^n, \forall w \in \mathbb{R}^n \mid \sum_{i=1}^{n} w_i = 1$$

$$s(y, x) = \max_{1 \leq i \leq n} w_i(y_i - x_i) + \epsilon \sum_{i=1}^{n} w_i(y_i - x_i)$$

$$\text{with } \epsilon > 0$$

In the previous expression we assume that y is the ideal point. The ideal point is the one which optimizes all the criterion individually. This point is not always a solution. By using this generator and fixing $\epsilon = 0$, any point x minimizing $s(x, y)$ is an efficient solution. To approximate the Pareto set we can take $\epsilon > 0$. The analytical formula of this function allow us to include it in our model as an additional constraint. We only have to introduce a variable to capture its value. Approximating the Pareto-frontier amounts to solve the NSCSP and to optimize the value of $s(x, y)$ for all values of w_i

This method has been tested on a mechanical model of lattice of beams. This model is described in the Xavier Fischer's Ph. D. thesis [Fis00]. The figure 5 shows the lattice system. The problem consists in determining what are the best beams (shape and material) that will ensure both a short motion of point A and a small weight of the entire system. The initial parameters are the angle ACB and ABC and a force applied on the top with a specific angle.

The constraint based model is composed of 32 variables with discrete and continuous domain, 20 hard constraints, 2 soft constraints and 1 generator of efficient solution. The two soft constraints express the preferences on the motion and on the weight. The Pareto-frontier is defined by the two criteria : weight and motion. This frontier represents the tradeoff between the two soft constraints. We use our transformation to obtain a NCSP. To solve it we use the solver of our implementation describe in section 5.1.

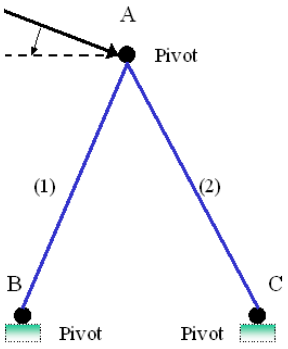

Fig. 5. The lattice of beam system. (1) and (2) are beams.

We chose this problem because we had already calculated the set of all solutions and from this set extracted the exact Pareto-frontier. Starting with the same model and using the generator we had approximated the Pareto-frontier. The results are summarized in the figure 6 which shows what should be the ex-

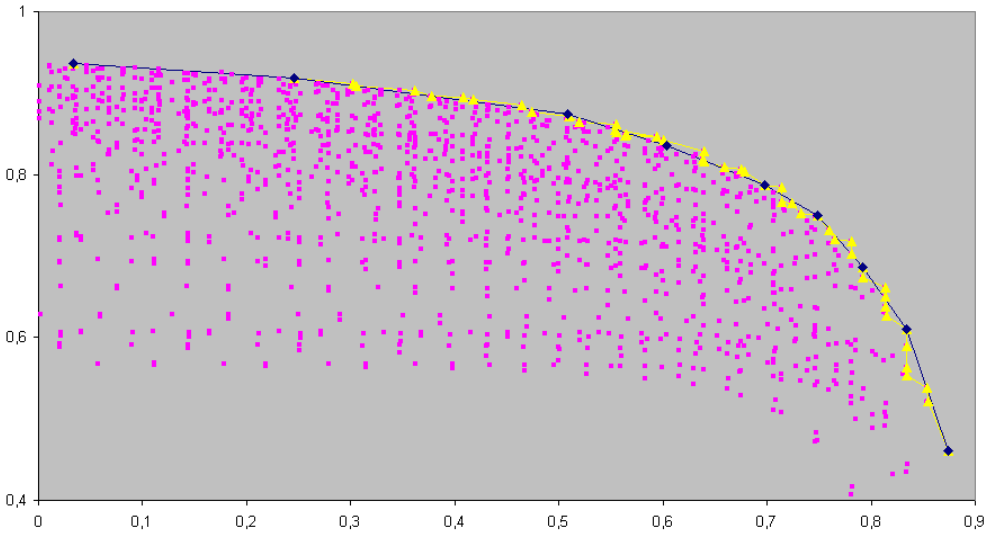

Fig. 6. Solutions of the lattice problem and Pareto-frontier. X axis represent the preferences on weight of the system. Y axis represent preferences on the motion of the lattice's top.

act Pareto-frontier and two successive approximations. The first approximation is represented by the black curve connecting the points symbolized by rhombuses. The white curve defined by the triangular points is obtained thanks to the second approximation.
These two approximations was generated by our generator in which we fix the value of w_i. The value of w_i goes from 0 to 1 with a step of 0.1 for the black curve, and a step of 0.01 for the white curve. The second approximation is obviously more precise but also more costly to compute. The method is conclusive. Even if we have, in the current implementation, to make by hand the successive optimizations for each step.

5 Implementation and results

5.1 Implementation

The DASSAULT-AVIATION's team "Contraintes et Décision" (Constraint and Decision) develops a software based on a NCSP solver: Constraint Explorer(CE). CE aims at modeling and solving design problems. This tool is the software used and enhanced in the French project CO2(an French acronym for constraint and design) granted by the French ministry of research .For a description of CO2 and CE see [Zim01]. The figure 7 presents a view of the interface of CE.
This software provides a mechanism for conditional constraint both in its lan-

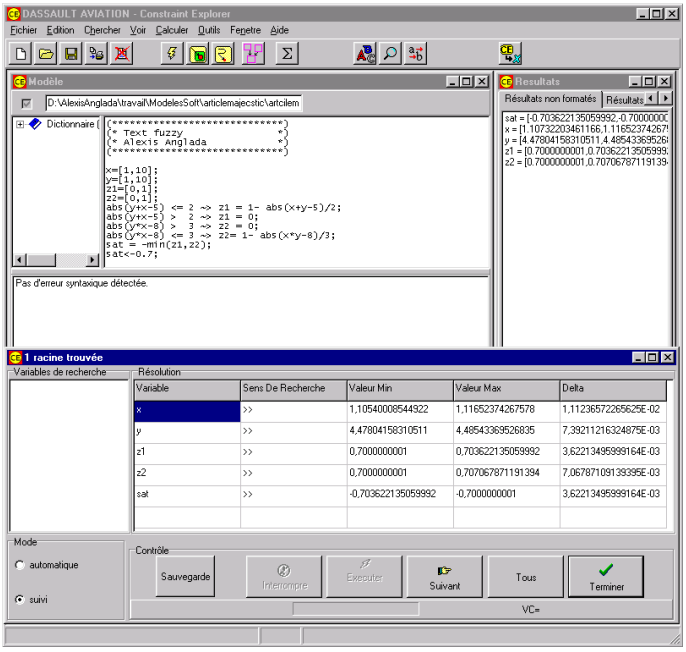

Fig. 7. Constraint Explorer's interface

guage and in its resolution algorithm. Hull consistency algorithm provides the propagation phase. Resolution is based on a backtrack algorithm with static ordering. Exploration of the search space is done by splitting domains with a dichotomy strategy. The optimization task is performed by a minimization algorithm. This algorithm reintroduces the last known solution as a higher boundary like in branch and bound algorithms. This scheme was derived from those summarized in [PM95]. It was adapted to the continuous field and implemented in CE ([Let01]).

5.2 Results

With our implementation we have solved the multicriteria problem. The results are given in the section 4.3.
To test our approach we have also solved the NSCSP of the section 4. We used the NCSP obtained by transformation in the section 4.1. We stopped our splitting strategy when domains reached a width less or equal to 1% of their low boundary. In a first time we have exhibited all the solutions and extracted the 3D curves of satisfaction show in figure 8. In a second time we have looked far the set of optimal solutions in sense of the FCSP paradigm. This set correspond to the top of the 3D curve and the value are detailed the table 1.

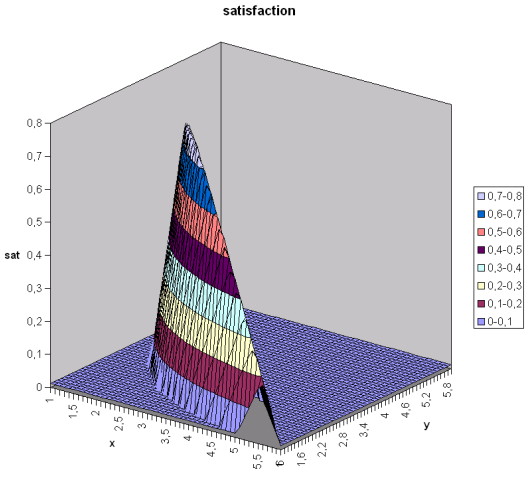

Fig. 8. Curve of total satisfaction of the solutions according to the values of the variables x and y.

Table 1. Optimal solutions

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

6 Conclusion and future work

In this paper, the Semiring based CSP theory has been extended to a numerical paradigm. We have defined a method to transform and solve Numerical Semiring based CSPs. The main issue of this work was to combine both numerical and soft features in order to fit with real problems.
We have also presented two graphical approaches allowing the user to express his/her satisfaction function. The first based on the piecewise linear curves and the second on the Béziers' curves.
Finally we have shown how to include multicriteria objectives in our framework.

We experimentally proved that NSCSPs can be solved by a NCSP solver with our transformation.
In the future, we want to compare our transformation with the abstracting method described in [BCR02].
Another investigation is to find some relevant semirings to express preferences and uncertainty in design.

References

[BCR02] Stefano Bistarelli, Philippe Codognet, and Francesca Rossi. Abstracting soft constraints: Framework, properties, examples. *Artificial Intelligence*, 139(2), 2002.

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

[BMR95] S. Bistarelli, U. Montanari, and F. Rossi. Constraint solving over semirings. In Chris Mellish, editor, *IJCAI'95 :Proceedings International Joint Conference on Atrtificial Intelligence*, pages 624–630, Montreal, 1995.

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

[BMR02] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Soft concurrent constraint programming. In *European Symposium on Programming*, pages 53–67, 2002.

[Béz68] Pierre Bézier. Procédé de définition des courbes et surfaces non mathématiques. *Automatisme*, XIII:189–196, 1968.

[Fis00] Xavier Fischer. Stratégie de conduite du calcul pour l'aide à la décision en conception mécanique intégrée ; application aux appareils à pression., 2000. Thèse de docteur en mécanique.

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

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

[SVF95] Thomas Schiex, Gérard Verfaillie, and Hélène Fargier. Valued constraint satisfaction problem : Hard and easy problem. In Chris Mellish, editor, *IJCAI'95 :Proceedings International Joint Conference on Atrtificial Intelligence*, pages 631–637, Montreal, 1995.

[Wie86] A.P. Wierzbicki. On the completeness and constructiveness of parametric characterizations to vector optimization problems. *OR Spektrum*, 8:73–87, 1986.

[Zim01] Laurent Zimmer. Presentation du projet CO2. In *S3P : Simulation de Produits, de Procédés et de Processus industriel*, France, 2001.