Specifying fuzzy constraints interactions without using aggregation operators

Joao Moura-Pires, Henri Prade

Specifying fuzzy constraints interactions without using aggregation operators

João Moura-Pires
Dep. de Informática - FCT
Universidade Nova de Lisboa
2825 Monte Caparica, Portugal
jmp@di.fct.unl.pt

Henri Prade
I.R.I.T., Université Paul Sabatier
118 route de Narbonne
31062 Toulouse Cedex 4, France
Henri.Prade@irit.fr

*Abstract-***A tuple of fuzzy constraints defined on a discrete satisfaction scale defines a lattice structure, called relaxation space, made of the tuples of level cuts of the fuzzy constraints, equipped with the partial order induced by the scale ordering. Then choosing a way of combining the fuzzy constraints has two joint effects: usually a completion of the ordering between tuples of levels, and possibly a simplification of the relaxation space (by eliminating subsumed tuples corresponding to the same level of global satisfaction). The paper investigates the possibility of specifying the ordering between tuples of level cuts constraints on a simplified relaxation space, without resorting to the use of explicit aggregation operations. The idea is to elicitate the preferences between the tuples of constraints representing various relaxations of the set of fuzzy constraints, directly from the user. How to express these preferences in a local manner is also discussed.**

I. Introduction

Constraints in a satisfaction problem are sometimes flexible and can be conveniently modelled by fuzzy sets for expressing preferences between more or less acceptable choices.

Fuzzy constraint satisfaction problems (FCSP) are an extension of constraint satisfaction problems (CSP), where elastic constraints can replace crisp ones. In a FCSP a constraint is satisfied to a degree (rather than satisfied or not satisfied), and the acceptability of a potential solution w.r.t. an aggregated set of fuzzy constraints becomes a gradual notion [15][3][6]. FCSPs are naturally encountered in different areas such as structural design [11], or scheduling [9] for instance.

In FCSPs, fuzzy constraints are usually aggregated by min operation. More recently, it has been shown [14][13] that any monotonically increasing aggregation operators can be dealt with, in the framework of a level cut representation of the fuzzy constraints. This is handled by introducing more nodes (than with min) in the relaxation space made of relaxed problems constituted by sets of level cuts of the fuzzy constraints, when a finite scale is used.

In this paper, we pursue the idea briefly suggested in [13] of taking advantage of the relaxation space [10] for expressing how the levels of satisfaction of the fuzzy constraints interact in the global satisfaction level attached to a solution of a FCSP, without necessarily referring to the explicit use of some aggregation operator. Indeed this interaction, which may be compensatory or not, can be directly expressed by simple requirements which complete the ordering in the relaxation space. It may lead to simplify the relaxation space, which is exploited for finding a solution to the FCSP.

The paper is organized as follows. Section 2 recalls how a fuzzy constraint can be viewed as a weighted collection of crisp constraints. Section 3 defines the relaxation space associated with a set of fuzzy constraints. Section 4 discusses min ordering and its refinements, discrimin and leximin, on the relaxation space. Section 5 analyzes how aggregation modes between fuzzy constraints are associated with further specifications of the ordering on the relaxation space and possible simplifications of this space. Section 6 suggests how the ordering and the simplifications on the relaxation space can be directly specified through requirements, given by the user. Section 7 briefly addresses some computational issues.

II. Fuzzy Constraints

A fuzzy constraint C over a set of variables is represented by a fuzzy set on the Cartesian product of the domains of the variables involved. The membership degrees express preferences among solutions of the constraint by ranking the instantiations, which are more or less acceptable for the satisfaction of this flexible constraint C.

Fuzzy sets are defined on a finite linearly ordered valuation set, denoted $L = \{\alpha_0 = \mathbf{0} < \alpha_1 < \ldots < \alpha_L = \mathbf{1}\}$, where $\mathbf{0}$ and $\mathbf{1}$ denote the bottom and top elements. The order-reversing map of the scale is still denoted by $1 - (\cdot)$.

A fuzzy constraint C can be viewed as a set of prioritized crisp constraints (see Fig. 1) [6]. Let $C_j = (C)_{\alpha_j} = \{x \mid \mu_C(x) \geq \alpha_j\}$ be the α_j-cut of C for $j = 1, L$. The priority attached to the crisp constraint C_j is $1 - \alpha_{j-1}$. C_1, the support of C, is the crisp constraint with the highest priority, say $1 = 1 - \alpha_0$; C_L, the core of C, has the lowest positive priority, $1 - \alpha_{L-1}$. A fuzzy constraint C expresses how C can be progressively relaxed. Letting $\mu_{C_j}(\boldsymbol{x}) = 1$ if $\boldsymbol{x} \in C_j$, and 0 otherwise, we have

$$\mu_C(\boldsymbol{x}) = \min_j \max (\alpha_j, \mu_{C_j}(\boldsymbol{x})). \quad (1)$$

A fuzzy constraint C is thus equivalent to the set of nested crisp constraints $C_1 \supseteq C_2 \supseteq \ldots \supseteq C_L$. $C_j = C_{j+1}$ is not forbidden. By convention, C_0 denotes the domain on which C is defined ($C_0 \supseteq C_1$). C_0 can be viewed as the most extreme relaxation of C, which is always satisfied.

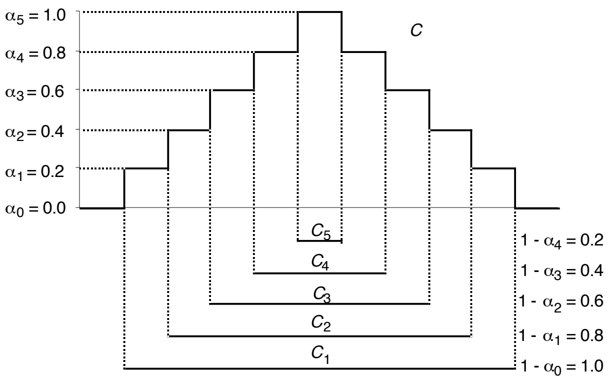


Fig. 1. α-cut view of a fuzzy constraint.

A fuzzy constraint C can be itself relaxed into C' by means of two elementary operations [13]:

- ρ–*importance weighting*:

$$\mu_{C'}(\boldsymbol{x}) = \max(1 - \rho, \mu_C(\boldsymbol{x})). \qquad (2)$$

It amounts to delete the most relaxed crisp constraints, from C_1 to C_k, where k is such that $\rho = 1 - \alpha_k$. In Fig. 1, for $\rho = 0.6$ ($\alpha_k = 0.4$ and thus $k = 2$) the two crisp constraints C_1 and C_2 are deleted.

- θ–*thresholding*:

$$\mu_{C'}(\boldsymbol{x}) = \theta \rightarrow \mu_C(\boldsymbol{x}), \qquad (3)$$

where a → b is Gödel implication (a → b = 1 if a ≤ b, a → b = b if a > b). It amounts to erase the hardest crisp representatives of C, from C_L to C_m, where m is such that $\theta = \alpha_{m-1}$. In Fig. 1, for $\theta = 0.8$ the crisp constraint C_5 is deleted.

III. Fuzzy Constraints as a Relaxation Space

Let $\{C^1, \ldots, C^K\}$ be a set of fuzzy constraints with their associated crisp representatives C^k_j (with priority 1 - α_{j-1}). It defines a relaxation space made of a lattice structure between K-tuples of crisp constraints ($C^1_{j_1}, \ldots, C^k_{j_k}, \ldots, C^K_{j_K}$) equipped with a partial order $<_c$ (defined below). For the sake of simplicity the K-tuple ($C^1_{j_1}, \ldots, C^k_{j_k}, \ldots, C^K_{j_K}$,) will be represented by the K-tuple $J = (j_1, \ldots, j_k, \ldots, j_K)$. Given two K-tuples $J = (j_1, \ldots, j_k, \ldots, j_K)$ and $I = (i_1, \ldots, i_k, \ldots, i_K)$, I is a relaxation of J, denoted by $I \leq_c J$, *iff* for all k from 1 to K, $\alpha_{i_k} \leq \alpha_{j_k}$, and I is said to be a *strict relaxation* of J, denoted by $I <_c J$, *iff* additionally there is a k such that $\alpha_{i_k} < \alpha_{j_k}$, and $C^k_{i_k} \supset C^k_{j_k}$. When $C^k_{j_k} = C^k_{i_k}$ for all k from 1 to K, then $I =_c J$, even if for some k, $j_k \neq i_k$ since $C_j = C_{j+1}$ is not forbidden.

Figure 2 shows the relaxation space associated with the set of three fuzzy constraints $\{A, B, C\}$ with grades ranging on level scale L = $\{\alpha_0 < \alpha_1 < \alpha_2 < \alpha_3\}$. This lattice structure reflects the tightness of the tuples of constraints. (A_3, B_3, C_3) made of the cores of the fuzzy constraints, is the hardest set of constraints, while (A_1, B_1, C_1), made of the supports, is the most relaxed set (where no constraint is completely forgotten).

It is worth stressing that it is not necessary that a relaxation space contain all possible K-tuples of level cuts of the fuzzy constraints. In fact, any subset of K-tuples of level cuts of the fuzzy constraints is a relaxation space. The number of nodes in the relaxation space is the result of the granularity required by the user.

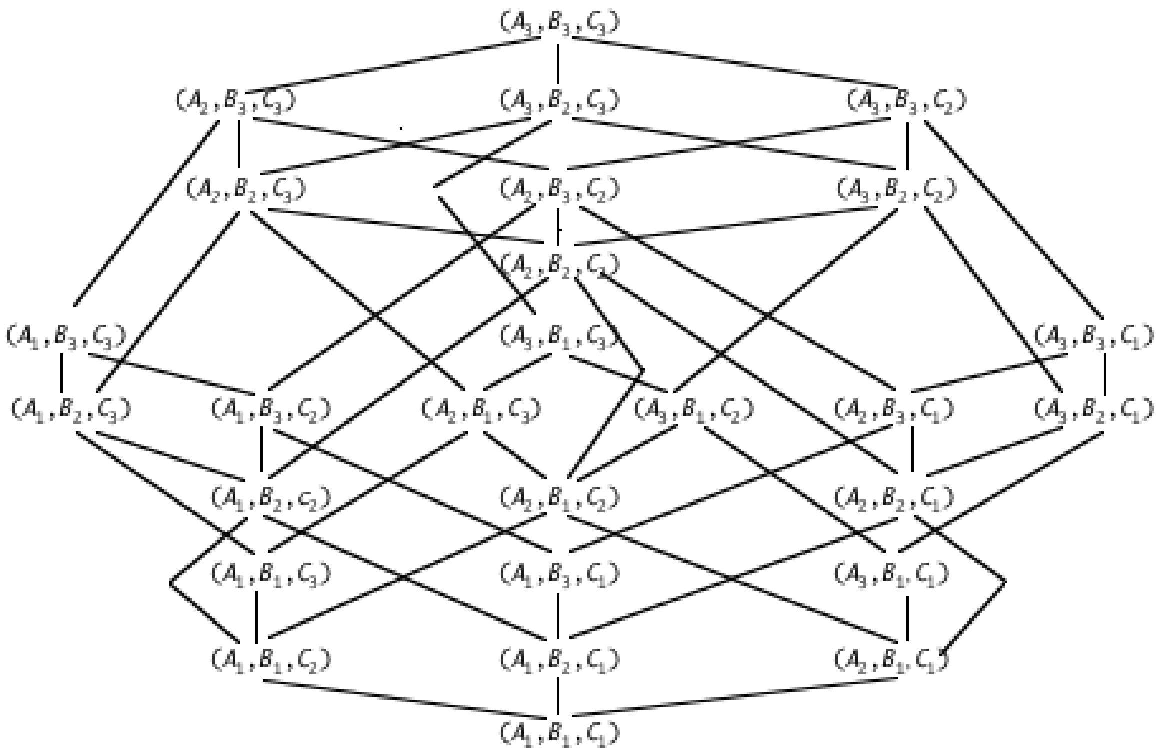


Figure 2. The relaxation space associated with the set of fuzzy constraints $\{A, B, C\}$.

Actually, the relaxation space shown in Fig. 2 is a subset of the largest relaxation space associated to the set of fuzzy constraints $\{A, B, C\}$. Indeed, none of the K-tuples shown in Fig. 2 includes one of the trivial relaxations A_0, B_0 or C_0. This provides a first example of specification that can be obtained from the user. Is (s)he interested in relaxing constraints beyond the support, is it permitted to forget some constraints? If the answer is no, the domains $C^1{}_0$, …, $C^K{}_0$ are not considered in the relaxation space.

IV. Fuzzy CSP and Orderings

A FCSP is defined by a set of decision variables X = $\{X_1, \ldots, X_N\}$, a set of domains D = $\{D_1, \ldots, D_N\}$ where D_i is the finite domain of X_i, and a set of fuzzy constraints C = $\{C^1, \ldots, C^K\}$where C^k is a fuzzy set defined over the Cartesian product of the domains of the variables involved in C^k. A membership value is associated to each tuple of values of the variables in C^k.

A solution to a FCSP, denoted by $\boldsymbol{x}$, is a vector of values in D_1 x … x D_N. The level of satisfaction of the constraints reached by $\boldsymbol{x}$, denoted by sat($\boldsymbol{x}$), is defined by [1]: $\text{sat}(\boldsymbol{x}) = \min_k \mu_{C^k}(\boldsymbol{x})$. It is the satisfaction degree of the least satisfied constraint. A *min optimal* solution is any $\boldsymbol{x}^*$ such that $\text{sat}(\boldsymbol{x}^*) = \max_{\boldsymbol{x}} \min_k \mu_{C_k}(\boldsymbol{x})$.

Since min-optimal solutions to a FCSP may be numerous, two refinements of the ranking of solutions have been proposed, the *discrimin* and *leximin orderings* [7][8], where not only the least satisfied constraint is taken into account, but the other satisfaction levels are also compared.

Discrimin ordering looks for the lowest satisfied constraint among the constraints that are not equally satisfied. Namely, let $\Delta(\boldsymbol{x}, \boldsymbol{y})$ be the subset of constraints that are not equally satisfied by $\boldsymbol{x}$ and $\boldsymbol{y}$, i.e., the C^k such that $\mu_{C^k}(\boldsymbol{x}) \neq \mu_{C^k}(\boldsymbol{y})$. Discrimin is a partial ordering defined by $\boldsymbol{x} >_{\text{disc}} \boldsymbol{y}$ *iff*

$$\min_{C^k \in \Delta(x,y)} \mu_{C^k}(\boldsymbol{x}) > \min_{C^k \in \Delta(x,y)} \mu_{C^k}(\boldsymbol{y}).$$

Let S_x be the fuzzy set of constraints satisfied by $\boldsymbol{x}$ which associates to each constraint C^k its satisfaction level $\mu_{C^k}(\boldsymbol{x})$. $(S_x)_\alpha = \{C_k \in C \mid \mu_{C_k}(\boldsymbol{x}) \geq \alpha\}$, the α-cut of S_x, is the set of constraints of C that are satisfied at least at level α. The discrimin ordering can be interpreted in terms of inclusion of α-cuts. It consists in comparing the α-cuts of S_x and of S_y, from the lowest level to the greatest one, until reaching a level α such that $(S_x)_\alpha \neq (S_y)_\alpha$. At this level if a strict inclusion $(S_x)_\alpha \supset (S_y)_\alpha$ holds, then $\boldsymbol{x} >_{\text{disc}} \boldsymbol{y}$. Otherwise, the solutions are incomparable.

The leximin ordering (e.g., [7]) is a refinement of min and discrimin orders. Let $\boldsymbol{s}_x$ be an *increasingly arranged* vector of the satisfaction degrees in S_x, i.e., $\boldsymbol{s}_x = (s_{[1]}, \ldots, s_{[K]})$, such that $s_{[i]} \leq s_{[i+1]}$ where $s_{[i]} = \mu_{C_{[i]}}(\boldsymbol{x})$ (the constraints are renumbered). Then, the leximin ordering is defined by:

$$\boldsymbol{x} >_{\text{leximin}} \boldsymbol{y} \Leftrightarrow \exists i \ \forall j < i \ \ s_{x_{[j]}} = s_{y_{[j]}} \text{ and } s_{x_{[i]}} > s_{y_{[i]}}.$$

This corresponds to a lexicographic comparison of the vectors $\boldsymbol{s}_x$ and $\boldsymbol{s}_y$. A solution $\boldsymbol{x}$ is leximin preferred to a solution ***y***, if there is a threshold α such that for all $\beta < \alpha$, the number of constraints satisfied by ***x*** at level at least β is equal to the number of constraints satisfied by ***y***, and ***x*** satisfies more constraints than ***y*** at level α.

While these orderings are originally defined over the set of FCSP solutions they are equivalently defined in terms of the K-tuples on the relaxation space. The leximin ordering provides a refinement of the relaxation order $<_c$. For instance, in Fig. 2, (A_3, B_3, C_2) is $<_c$-incomparable to $\{A_2, B_2, C_3\}$ while $\{A_3, B_3, C_2\}$ >-leximin $\{A_2, B_2, C_3\}$. It also induces ties: $\{A_2, B_2, C_3\}$, $\{A_2, B_3, C_2\}$, $\{A_3, B_2, C_2\}$ are leximin equivalent. With the leximin refinement all the tuples of constraints become comparable. This is not the case with discrimin, which offers a less strong refinement. For instance $\{A_3, B_3, C_2\}$ and $\{A_2, B_2, C_3\}$ remain discrimin-incomparable, while $\{A_2, B_1, C_3\}$ is discrimin preferred to $\{A_3, B_1, C_1\}$ although they are $<_c$-incomparable.

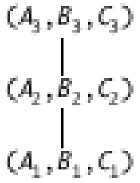


Fig. 3. Relaxation space of min-FCSP.

These two refined orderings take place on the same relaxation structure. The relaxation space associated with the *min* ordering is pictured in Figure 3. Note that it is a very simplified version of the one defined by $<_c$ in Fig. 2. It appears that two phenomena are at work: suppressing incomparabilities on the one hand, and simplifying the relaxation space on the other hand. Clearly, there are many ways to get rid of incomparabilities (in agreement with $<_c$). Besides, it may be desirable to decouple the simplification of the relaxation space and the suppression of incomparabilities (which are mixed together by the use of the min operation).

It has been shown that the result of any monotonic combination of fuzzy constraints can be encoded as a set of prioritized crisp constraints, which can be syntactically obtained from the initial ones [14][13].

This includes noticeable particular cases such as the product, or the average of fuzzy constraints. This also allows for the use of different operations for combining more than two constraints (e.g., a weighed average of *A* with the min combination of *B* and *C*), thus modeling different types of interactions between constraints such as tradeoffs or pure conjunctive aggregations. However, such an approach has some limitations although it can accommodate any practical aggregation mode. First, as in the case of min, the simplification of the relaxation space and the suppression of incomparabilities cannot be controlled separately.

Second, it is not always easy to capture all the possible aggregation modes by means of a set of combination operations. In the following, a more flexible way of specifying the interactions between fuzzy constraints is proposed.

V. Implicit Aggregation in the Relaxation Space

In this section we consider the case of the interaction between two fuzzy constraints *A* and *B* defined using the same scale $L = \{\alpha_0 = \mathbf{0} < \alpha_1 < \ldots < \alpha_L = \mathbf{1}\}$. We assume that if some thresholding or importance weighting takes place, the result of these elementary modifications is already integrated in the constraint. As already said the relaxation space does not specify all the preferences between pairs of crisp constraints (A_i, B_j) and (A_k, B_l). If A_i (resp. B_j) is satisfied, at least the satisfaction level α_i (resp α_j) is reached. What $<_c$ reflects is a partial Pareto order on the pairs of valuations (α_i, α_j), namely $(\alpha_0, \alpha_0) <_c (\alpha_0, \alpha_1) <_c (\alpha_1, \alpha_1) <_c \ldots <_c (\alpha_{L-1}, \alpha_L) <_c (\alpha_L, \alpha_L)$. For notational simplicity we shall write (0, 0) < (0, 1), etc. Since the constraints are assumed to play symmetrical roles, pairs (i, j) and (j, i) are equivalent and by convention when we write (i, j) it is assumed that $i \leq j$. More generally, we have $(i, j) < (k, l)$ as soon as $i \leq k$ and $j < l$, or $i < k$ and $j \leq l$. What remains to specify is the ordering between pairs (i, j) and (k, l) such that $i > k$ and $j < l$.

The situation is pictured in Fig. 4, for two constraints, and $L = \{\alpha_0 = \mathbf{0} < \alpha_1 < \alpha_2 < \alpha_3 = \mathbf{1}\}$. The orderings remaining to specify are pictured in dotted lines. Such a refinement of $<_c$ will be denoted by $<_p$ (preference ordering).

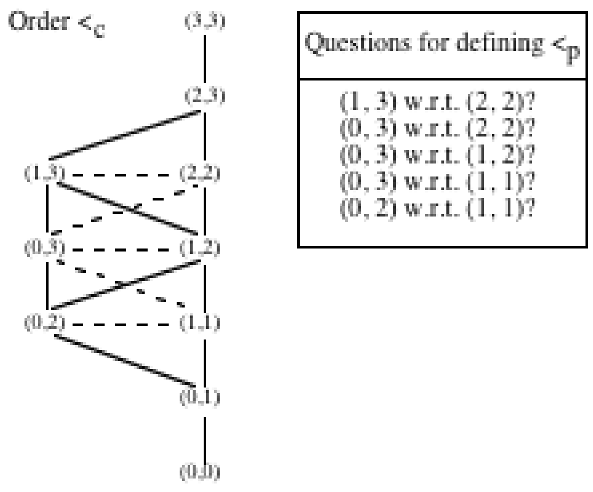
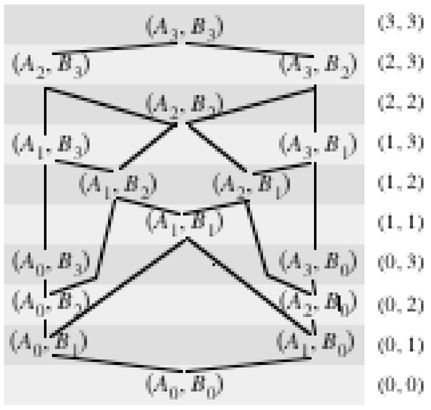


Fig. 4. Refinements of Pareto ordering .

For instance, from Fig. 4, (1, 3) $<_c$ (2, 3) and (2, 2) $<_c$ (2, 3), while (2, 2) is $<_c$-incomparable to (1, 3) and to (0, 3). Moreover, the specifications should obey the transitivity requirement, for instance the user cannot enforce both (1, 3) $<_p$ (2, 2) and (2, 2) $<_p$ (0, 3). Additionally the user may consider some pairs as equally good. For instance, (0, 2) $=_p$ (1, 2), or even (2, 3) $=_p$ (2, 2). Such equalities can eventually simplify the relaxation space as we will explain later.

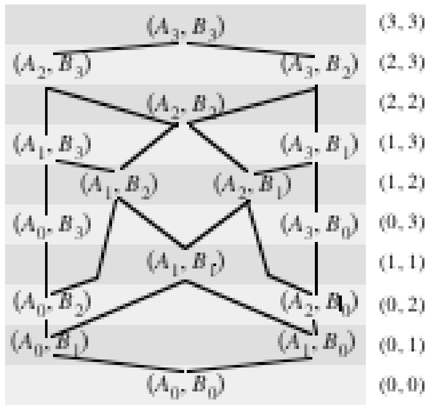
Figure 5 exhibits the leximin linearization of $<_c$. Note that it corresponds to the following requirements: *a*) (2, 2) $>_p$ (1, 3) (which implies that (2, 2) $>_p$ (0, 3)); *b*) (1, 2) $>_p$ (0, 3) (which could be deduced from the previous requirement assuming similar preferences along the satisfaction scale); *c*) (1, 1) $>_p$ (0, 3) (which implies that (1, 1) $>_p$ (0, 2)).



(A_3, B_3) (3, 3)
(A_2, B_3) (A_3, B_2) (2, 3)
(A_2, B_2) (2, 2)
(A_1, B_3) (A_3, B_1) (1, 3)
(A_1, B_2) (A_2, B_1) (1, 2)
(A_1, B_1) (1, 1)
(A_0, B_3) (A_3, B_0) (0, 3)
(A_0, B_2) (A_2, B_0) (0, 2)
(A_0, B_1) (A_1, B_0) (0, 1)
(A_0, B_0) (0, 0)

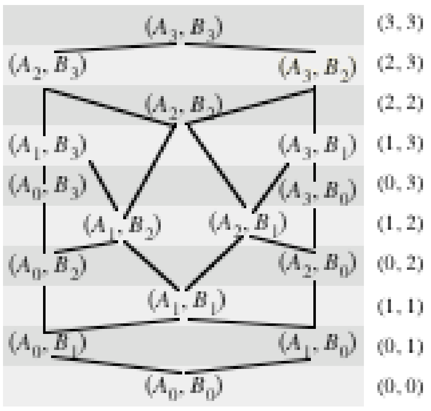
Fig. 5. Leximin.

Figure 6 shows a different linearization, which can be interpreted as an average refined by the min. The necessary requirements to reach this linearization are similar to the previous ones except for the requirement *c*) replaced by *c*'): (0, 3) $>_p$ (1, 1) and the additional requirement, *d*) (1, 1) $>_p$ (0, 2). Figure 7 shows a possible combination mode, which is more difficult to interpret in terms of known aggregation operators. However, it can be specified using simple requirements as in the previous cases: *a*) (2, 2) $>_p$ (1, 3) (which implies that (2, 2) $>_p$ (0, 3)); *b*) (0, 3) $>_p$ (1, 2) (which implies that (0, 3) $>_p$ (1, 1)); *c*) (0, 2) $>_p$ (1, 1) (which could be deduced from the previous requirements assuming similar preferences along the satisfaction scale).



(A_3, B_3) (3, 3)
(A_2, B_3) (A_3, B_2) (2, 3)
(A_2, B_2) (2, 2)
(A_1, B_3) (A_3, B_1) (1, 3)
(A_1, B_2) (A_2, B_1) (1, 2)
(A_0, B_3) (A_3, B_0) (0, 3)
(A_1, B_1) (1, 1)
(A_0, B_2) (A_2, B_0) (0, 2)
(A_0, B_1) (A_1, B_0) (0, 1)
(A_0, B_0) (0, 0)

Fig. 6. Average refined by min.



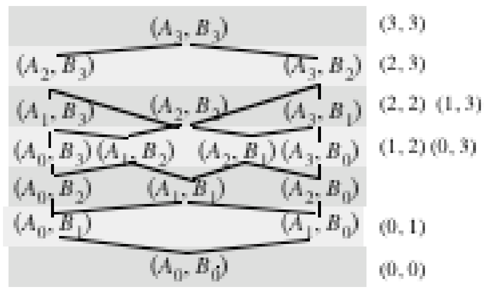
(A_3, B_3) (3, 3)
(A_2, B_3) (A_3, B_2) (2, 3)
(A_2, B_2) (2, 2)
(A_1, B_3) (A_3, B_1) (1, 3)
(A_0, B_3) (A_3, B_0) (0, 3)
(A_1, B_2) (A_2, B_1) (1, 2)
(A_0, B_2) (A_2, B_0) (0, 2)
(A_1, B_1) (1, 1)
(A_0, B_1) (A_1, B_0) (0, 1)
(A_0, B_0) (0, 0)

Fig. 7. A non classical aggregation.

It is worth stressing that in the three previous examples $<_p$ has no ties and the number of global satisfaction levels uses a scale with 10 elements (from (0, 0) to (3, 3)). There are 12 different linearizations of $<_c$ without ties for two constraints and L = 3, and the relaxation space is always the same, differing only on the preferred nodes.

If we relax the requirement of strict preference specification and allows ties then other combinations would be obtained. Note that these other combination schemes take place on other relaxation spaces, which are simplified versions of the initial one. Whenever the user defines equalities between $<_c$-comparable tuples the relaxation space becomes more simplified, i.e., some relaxation graph nodes are removed .

Figure 8 shows the relaxation space corresponding to the average aggregation. Here, the average is not refined by the min (or by any other way) and ties are exhibited. Suppose that (A_1, B_3), (A_2, B_2) and (A_3, B_1) are maximally consistent. While in the case of Fig. 6 (A_2, B_2) would be the preferred one, here any of those three pairs are considered equally satisfactory.



(A_3, B_3) (3, 3)
(A_2, B_3) (A_3, B_2) (2, 3)
(A_1, B_3) (A_2, B_2) (A_3, B_1) (2, 2) (1, 3)
(A_0, B_3) (A_1, B_2) (A_2, B_1) (A_3, B_0) (1, 2) (0, 3)
(A_0, B_2) (A_1, B_1) (A_2, B_0)
(A_0, B_1) (A_1, B_0) (0, 1)
(A_0, B_0) (0, 0)

Fig. 8. Average.

Figure 9 exhibits a complete order, which can be viewed as an aggregation by the product. Note that in this case the number of relaxation space nodes is smaller than in the previous examples. In fact due to the ties from (0, 0) to (0, 3) the relaxation space is simplified by keeping only the more relaxed pair(s) from the pairs within a tie.

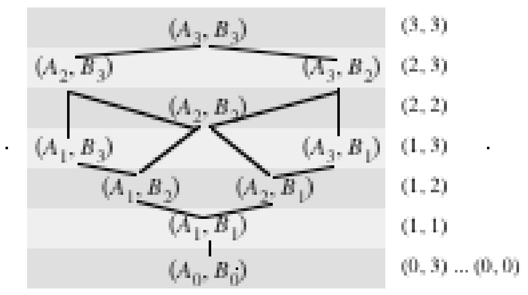


Fig. 9. Product.

It is worth stressing that simplified relaxation spaces can also be specified without the explicit use of aggregation operators. Figure 10 shows a relaxation space which is specified through the following requirements: *a*) (1, 3) $>_p$ (2, 2); *b*) it is not acceptable to relax any constraints beyond its support.

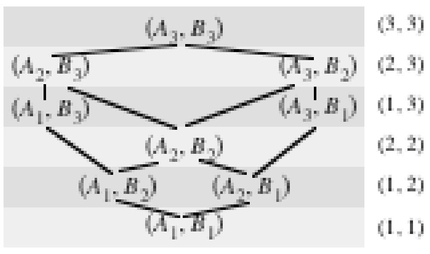


Fig. 10. A simplified specified relaxation space.

VI. Specification of Constraint Interactions

The previous section has shown, in the particular case of the combination of two fuzzy constraints, how the refinements of the ordering on the relaxation space and the simplification of the space are reflecting aggregation modes. Indeed, we can directly take advantage at the computational level of the specification of any refinement of the ordering $<_c$ on the relaxation space, or of the simplification of this space. Simplifying the relaxation space enforces that tuples of level cut constraints (constituting crisp representatives of the original fuzzy constraint satisfaction problem) have an equal level of satisfaction from the user point of view, and amounts to eliminate subsumed tuples (just keeping the easiest set of constraints). It may be interesting to start with a rather simplified relaxation space, and depending on what set of constraints in this space reveals to be feasible or infeasible, to later specify a partial refinement of the ordering leading to a local unfolding of the relaxation space, once a maximal consistent satisfaction problem has been identified.

Let us now suggest how requirements on the ordering between the tuples of crisp level cut constraints can be expressed. While it is easy to express the relative position in the ordering of e.g., (0,2) with respect to (1,1), it may be much more tedious to express the direct assessment of orderings between *K*-tuples with *K* larger than 2 or 3. What might be more natural is to reconstruct the ordering from partial comparisons of levels of satisfaction of small subsets of constraints. This will also enable the expression of tradeoffs between some constraints, while others are combined in a pure conjunctive style for instance.

Generally speaking, a (partial) ordering, refining $<_c$, between *K*-tuples $(\alpha_1, \ldots, \alpha_K)$, where $\alpha_k \in$ L, can be specified through different types of requirements:

- (partial) examples of preferences such as, e.g., $(\alpha_1, \ldots, \alpha_{k-1}, 0, \ldots, \alpha_{m-1}, 2, \ldots, \alpha_K)$ is not preferred to $(\alpha_1, \ldots, \alpha_{k-1}, 1, \ldots, \alpha_{m-1}, 1, \ldots, \alpha_K)$ for any pair $k \neq m$;
- preferences between reduced tuples of global satisfaction levels of subsets of constraints, assuming that each subset is aggregated using some standard combination operation;
- generic preferences expressing, for instance, statements about the relative importance of criteria, e.g., "*k* is as important *m*": $(\ldots., \alpha_k, \ldots, \alpha_m, \ldots) \cong_p (\ldots., \alpha_m, \ldots, \alpha_k, \ldots)$ for all α_k and α_m, the other α_i being identical in both tuples; "*k* is more important than *m*" $(\ldots., \alpha_k, \ldots, \alpha_m, \ldots) >_p (\ldots., \alpha_m, \ldots, \alpha_k, \ldots)$ for $\alpha_k > \alpha_m$.
  - generic statements expressing preferential independence between sets of criteria **I** and $\mathbf{I}^c$, i.e., $(\alpha_\mathbf{I}, \alpha_{\mathbf{I}^c}) >_p (\beta_\mathbf{I}, \alpha_{\mathbf{I}^c}) \Leftrightarrow (\alpha_\mathbf{I}, \beta_{\mathbf{I}^c}) >_p (\beta_\mathbf{I}, \beta_{\mathbf{I}^c})$ where **I** is a subset of $\{1, \ldots, K\}$ and $\mathbf{I}^c$ its complement and $\alpha_\mathbf{I}$ stands for the vector of the α_i 's where $i \in \mathbf{I}$.

All such requirements contribute to a partial specification of an ordering complementing $<_c$. The consistency of requirements can be checked. Requirements and simplifications can be expressed progressively as mentioned at the beginning of this section. Note also that requirements provided through examples might be generalized, i.e., extrapolated into generic requirements. For instance if we state that $(\ldots, 0, \ldots, 2, \ldots) >_p (\ldots, 1, \ldots, 1, \ldots)$, it might be extrapolated into $(\ldots, 0 + x, \ldots, 2 + x, \ldots) >_p (\ldots, 1 + x, \ldots, 1 + x, \ldots)$.

VII. Conclusions: Computational Issues

This paper advocates the idea that preferences between potential solutions of a set of fuzzy constraints can be expressed through examples based on generic requirements, which can be directly exploited on the relaxation space, rather than through the use of explicit aggregation operations. This should enables a more flexible and local representation of preferences, as also recently discussed in [2] in a multiple criteria aggregation setting.

The control of the relaxation space through the requirements, which are entered by the user, has also some computational interest. Given a relaxation space the user intends to find one or all $<_c$-maximal consistent *K*-tuples in the relaxation space. Note that all maximal consistent problems are $<_c$-incomparable. Once a refinement $<_p$ is provided the task is then to find one or all $<_p$-maximal consistent problems, which are generally less than the number of $<_c$-maximals. In general, the determination of consistency of a set of crisp constraints is a NP-complete problem [12]. Solving a min-FCSP, whose relaxation space is pictured in Fig. 3, amounts to solve, in the worst case, *L* sets of crisp constraints, *L* being the number of positive levels of the satisfaction scale. Min-optimal solutions for FCSP can be obtained by solving a logarithmic number of CSP's [4].

The flexibility which is introduced in min-FCSPs has a computational price that is controllable by the user, through the number of satisfaction levels in the satisfaction scale. In other words, a greater granularity of the relaxation space provides a greater flexibility at the expense of some computational price.

The idea of a tradeoff between granularity and computational price can be considered for any relaxation space. The hardness of finding one $<_p$-best maximal consistent *K*-tuple in the relaxation space grows with the number of nodes in the relaxation space.

As said before, ties in $<_p$ can produce simplified versions of the relaxation spaces. Starting with a preference relation $<_p$ with many ties will provide a small relaxation space, for which a solution can be found. Successive redefinitions of $<_p$ breaking ties will locally unfold the relaxation space and will provide solution refinements. Another way to produce simplified relaxation spaces is by θ–thresholding all the fuzzy constraints. Similar ideas for solving Valued-CSP by successive approximations can be found in [5].

REFERENCES

[1] R.E Bellman, and L.A. Zadeh, "Decision-making in a fuzzy environment", 1970, *Manag. Sc.*, 17, 141-164.

[2] S. Benferhat, D.Dubois. H.Prade (1999), Toward a possibilistic logic handling of preferences. *Proc. 16th Int. Joint Conf. in Artificial Intelligence* (IJCAI-00), Stockholm, 2000. 1370-1375.

[3] J. Bowen, R. Lai, and D. Bahler, "Fuzzy semantics and fuzzy constraint networks", *Proc. of the National Conf. on Artificial Intelligence (AAAI'92)*, San Francisco, 1992. 1009-1016.

[4] M Cooper, "Computational complexity of fuzzy constraint satisfaction", Tech. Report, Univ. of Toulouse, France, 1994.

[5] DeGivry, S. and G. Verfaillie. "Optimum anytime bounding for constraint optimization problems." in Proc. of the AAAI'97, Workshop on "Building Resource-Bounded Reasoning Systems", Providence, Rhode Island, USA, 1997.

[6] D. Dubois, H. Fargier, and H. Prade, "Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty", *Applied Intelligence*, 6, 1996, 287-309.

[7] D. Dubois, H. Fargier, and H. Prade, "Refinements of the maximin approach to decision making in a fuzzy environment", *Fuzzy Sets and Syst.*, 81, 1996, 103-122.

[8] D. Dubois, H. Fargier, and H. Prade, "Beyond min aggregation in multicriteria decision: (Ordered) weighted min, discri-min, leximin", in *The Ordered Weighted Averaging Operators — Theory and Applications* (R.R. Yager, J. Kacprzyk, eds.), Kluwer Publ., 1997, 181-192.

[9] H. Fargier, "Fuzzy scheduling: principles and experiments", in *Fuzzy Information Engineering: A Guided Tour of Applications* (D. Dubois et al., eds.), Wiley, New York, 1997, pp. 655-668.

[10] E. C. Freuder, "Partial constraint satisfaction", *Proc. of the Inter. Joint Conf. on Artificial Intelligence (IJCAI'89)*, Detroit, 1989, 278-283.

[11] Q. Guan, and G. Friedrich, "Extending constraint satisfaction problem solving in structural design", *Proc. of the 5th Inter. Conf. IEA/AIE*, Paderborn, 1992, LNAI nº 604, Springer Verlag, 341-350.

[12] Mackworth, A., "Consistency in networks of relations." *Artificial Intelligence* 8, 1977, 99-118.

[13] Moura-Pires, J., Dubois, D. and Prade, H, "Fuzzy constraint problems with general aggregation operations under possibilistic logic form, *Proc of 6th Europ. Cong. on Intelligent Techniques and Soft Computing* (EUFIT-98), Aachen, Germany Sept 7-10, 1998, 535-539.

[14] Moura-Pires, J. and Prade, H, "Logical analysis of fuzzy constraint satisfaction problems" *Proc. of the 7th IEEE Int. Conf. on Fuzzy Systems*, Anchorage, May 5-9, 1998, 857-862.

[15] K. Satoh, "Formalizing soft constraint by interpretation ordering", *Proc. of the Europ. Conf. on Artificial Intellig. (ECAI'90)*, Stockholm, 1990, 585-590.