INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center St. • Suite 600 • Berkeley, California 94704-1198 • (510) 643-9153 • FAX (510) 643-7684

A Formal Framework for Weak Constraint Satisfaction Based on Fuzzy Sets

Hans Werner Guesgen*

TR-94-026

June 1994

Abstract

Recent work in the field of artificial intelligence has shown that many problems can be represented as a set of constraints on a set of variables, i.e., as a constraint satisfaction problem. Unfortunately, real world problems tend to be inconsistent, and therefore the corresponding constraint satisfaction problems don't have solutions. A way to circumvent inconsistent constraint satisfaction problems is to make them fuzzy. The idea is to associate fuzzy values with the elements of the constraints, and to combine these fuzzy values in a reasonable way, i.e., a way that directly corresponds to the way how crisp constraint problems are handled.

*On leave from the Computer Science Department at the University of Auckland, Private Bag 92019, Auckland, New Zealand, email: hans@cs.auckland.ac.nz. The author has been supported by the University of Auckland Research Fund under the grant numbers A18/XXXXX/62090/3414014 and A18/XXXXX/62090/F3414025. Many thanks to Joachim Hertzberg for discussing the concepts described in this paper.

1 Introduction

Constraint satisfaction is a valuable technique for solving problems that can be stated in the following way: Given variables $\{V_1, \ldots, V_m\}$ over some domains $\{D_1, \ldots, D_m\}$ and constraints $\{C_1, \ldots, C_n\}$ with variables in $\{V_1, \ldots, V_m\}$ (i.e., each C_i is a pair consisting of a subset of the variables $\{V_1, \ldots, V_m\}$ and a relation on this subset), find a tuple of values $(d_1, \ldots, d_m) \in D_1 \times \cdots \times D_m$ such that all constraints are satisfied if d_i is assigned to V_i, $i = 1, \ldots, m$.

A real world problem is likely to be inconsistent, i.e., the corresponding constraint satisfaction problem (CSP) does not have a solution. If this is the case, one cannot satisfy the entire set of constraints but has to "weaken" some of the constraints in order to find a solution. Weakening constraints is usually achieved by replacing the relations of the constraints with supersets of their relations, hoping that the resulting, new CSP has a solution, which in turn might be viewed as an approximate solution or quasi-solution of the original problem. This process is called constraint relaxation.

There are several approaches to constraint relaxation. In [12, 7], some theoretical formulations of constraint relaxation are presented. [2, 5] describe practical applications in which constraint relaxation is used. All these approaches have in common that they attack a given CSP by finding a solution of a relaxed CSP that differs only minimally from the original CSP. The difference is expressed in terms of a metric.

There are various ways to define a metric on a given CSPs, i.e., to state how far a relaxed CSP is from the original CSP and with this how far away the approximate solution is from the ideal one. In [9], for example, we have used the concept of penalties: Values not being in the original constraint relations are marked by natural numbers greater than 0. More recently, fuzzy set theory has been used to capture the idea of constraint relaxation [3].

The main idea of [3] is to switch off constraints, i.e., to ignore them in the constraint networks, if they can't be satisfied. Some constraints can be ignored more easily than others. To express how easy it is to ignore a constraint, each constraint C is associated with a priority degree α_C ranging in the scale $[0, 1]$. $1 - \alpha_C$ then indicates to what extend it is possible to violate C.

Given a network of constraints with priorities, an assignment of values $(d_1, \ldots, d_m)$ to the variables $\{V_1, \ldots, V_m\}$ of the constraint network can be associated with some fuzzy membership grade. This membership grade is computed from the priorities of the constraints. If a constraint C is not satisfied by $(d_1, \ldots, d_m)$, then $1 - \alpha_C$ limits the membership grade of the solution.

The approach presented in this paper is very close to the one in [3]. The main difference is that we don't associate a fixed priority with each constraint of the network but define a constraint as being more or less satisfied by some given assignment of values. For example, instead of stating *The color of the object is supposed to be red* with priority 0.75 and then returning a membership grade of 1 if the object is indeed red or $1 - 0.75 = 0.25$ if the object is not red, we proceed as follows: There is a constraint *The color of the object must be red* which has the same priority as the other constraints in the networks. However, this constraint may be more or less satisfied. If the color of the object is red, then the constraint is satisfied with degree 1; if the color of the object is burgundy, then the constraint might be satisfied with degree 0.75, if the color is pink, then with degree 0.5; and so on.

In other words, we are assigning membership grades to the elements of the extended constraint relation, each membership grade specifying how far away this element is from the intended relation. A value of 1 means that the element has been in the original relation; a value of 0 means that the element is not admissible as an approximation. The question, of course, is how to combine the different membership grades into a single membership grade. This question is discussed in the next section.

2 Combining Membership Grades

The main operations performed on the relations represented by a constraint network are union and intersection: A constraint may be viewed as union of one-element sets, each element in such a set representing a possible choice of values for the variables of the constraint, i.e., an assignment that satisfies the constraint. A constraint network, on the other hand, may be viewed as intersection of the relations represented by the constraints of the network. Therefore, we have to look at the operations of union and intersection in the light of fuzzy set theory.

Several combination schemes have been introduced in the literature on fuzzy set theory for computing the union and intersection of fuzzy sets (for example, [13, 14, 1, 4]). The original min/max combination scheme introduced by Zadeh [15] seems to be the most adequate for fuzzy constraints, as it satisfies the following requirements:

No multiple reduction of grades

Suppose there are constraints in the network with identical variables and relations. Then the result of combining these constraints is supposed to be the same as the result obtained if there were only one of the constraints in the network. This means we don't want to accumulate membership grades.

No accumulation of grades

If an element occurs twice in a relation of a constraint, then the combination rule applied to this constraint should yield the same result as if the element occurs only once, i.e, we don't want to accumulate membership grades. This is certainly not an issue from the set-theoretical point of view, but may play a role in practical applications. For example, the constraint satisfaction system CONSAT [8] allows relations to be specified by patterns, where a relation element may be specified by more than one pattern.

Free choice of distributing relations

If a given relation is distributed over two different sets of constraints, then the combination rules should produce identical results for both sets of constraints. This means that it doesn't matter how a relation is represented in a constraint network.

With these requirements in mind, we can now define a formal framework for fuzzy constraints.

3 A Formal Framework

The main idea of fuzzy sets is: Given a classical set of elements, D, instead of specifying whether an element d belongs to a set $D' \subseteq D$ or not, we assign a membership grade to d. Formally, this is captured in the following definition:

Definition 1 (Fuzzy Set)
Let D be a classical (crisp) set, i.e., a collection of elements d, then a fuzzy set $\tilde{D}$ in D is a set of ordered pairs

$$\tilde{D} = \{\langle d, \mu_{\tilde{D}}(d)\rangle \mid d \in D\}$$

$\mu_{\tilde{D}}(d)$ is called the membership function or membership grade of d in $\tilde{D}$. It maps D to the membership space $M \subseteq \mathrm{I\!R}_0^+$, the supremum of M being finite:

$$\mu : d \in D \longmapsto m \in M$$

If $M = [0, 1]$, then $\mu_{\tilde{D}}(d)$ is called normalized.

The membership function replaces the characteristic function of classical sets, which map a given set D to $\{0, 1\}$ and thereby indicating whether an element belongs to D (indicated by 1) or not (indicated by 0). If $M = \{0, 1\}$, $\tilde{D}$ is nonfuzzy and $\mu_{\tilde{D}}(d)$ is identical to the characteristic function of a nonfuzzy set.

To handle constraints with arbitrary arity, we have to extend this definition to fuzzy relations. The first idea might be to define a fuzzy relation just as a simple fuzzy set in the product space of the underlying sets. However, this isn't sufficient, since we want to guarantee that the membership grade of an element of the relation does not exceed the membership grade of each component of the element. We therefore define:

Definition 2 (Fuzzy Relation)
Let $D_1, \ldots, D_k \subseteq D$ be classical sets and $\tilde{D}_1, \ldots, \tilde{D}_k$ fuzzy sets in $D_1, \ldots, D_k$, respectively. Then

$$\tilde{R} = \{\langle (d_1, \ldots, d_k), \mu_{\tilde{R}}(d_1, \ldots, d_k)\rangle \mid (d_1, \ldots, d_k) \in D_1 \times \cdots \times D_k\}$$

is a fuzzy relation on $\tilde{D}_1, \ldots, \tilde{D}_k$ in the product space $D_1 \times \cdots \times D_k$ if $\forall (d_1, \ldots, d_k) \in D_1 \times \cdots \times D_k$ and $\forall i \in \{1, \ldots, k\}$

$$\mu_{\tilde{R}}(d_1, \ldots, d_k) \leq \mu_{\tilde{D}}(d_i)$$

A constraint is usually defined as a pair consisting of a set of variables and a relation on these variables. To adopt this definition for fuzzy constraints, we have to define the notion of a fuzzy variable:

Definition 3 (Fuzzy Variable)
A fuzzy variable $\tilde{V}$ *in* D *is a variable that may range over any fuzzy set* $\tilde{D}$ *in* D. D *is called the domain of* $\tilde{V}$.

With the above definitions, we can now define the notion of a fuzzy constraint in a straightforward way. Actually, it is the direct analog of how classical (crisp) constraints are defined:

Definition 4 (Fuzzy Constraint)
A k*-ary fuzzy constraint* $\tilde{C}$ *is a pair* $\langle(\tilde{V}_1, \ldots, \tilde{V}_k), \tilde{R}\rangle$ *for which the following holds:*

1. $\tilde{V}_1, \ldots, \tilde{V}_k$ *are fuzzy variables in some given domains* $D_1, \ldots, D_k$*, respectively.*
2. $\tilde{R}$ *is a fuzzy relation in the product space* $D_1 \times \cdots \times D_k$.

Usually, we are dealing with a set of variables and a set of constraints on subsets of the variables. The constraints may interfere with each other in that they are using variables in common. As a result, we get a constraint network, or in the case of fuzzy sets, a fuzzy constraint network:

Definition 5 (Fuzzy Constraint Network)
An fuzzy constraint network $\tilde{N}$ *is a pair consisting of a set of fuzzy variables* $\{\tilde{V}_1, \ldots, \tilde{V}_m\}$ *and a set of fuzzy constraints* $\{\tilde{C}_1, \ldots, \tilde{C}_n\}$*, the variables of each* $\tilde{C}_i$ *being a subset of* $\{\tilde{V}_1, \ldots, \tilde{V}_m\}$.

With these definitions we have the means to express noncrisp knowledge in a fuzzy constraint network. The question is how do we perform inferences on such a network of fuzzy constraints. This question will be addressed in the following.

Classical constraint satisfaction algorithms are mainly based on the operations of union and intersection of relations. To be able to apply the classical constraint satisfaction algorithms to fuzzy constraint networks, we must define these operations for fuzzy relations:

Definition 6 (Union and Intersection)
Let $\tilde{R}$ *and* $\tilde{R}'$ *be two fuzzy relations in the same product space* $D_1 \times \cdots \times D_m$*, then the union and intersection of* $\tilde{R}$ *with* $\tilde{R}'$ *are defined by*

$$\mu_{\tilde{R}\cup\tilde{R}'}(d_1, \ldots, d_m) = \max\{\mu_{\tilde{R}}(d_1, \ldots, d_m), \mu_{\tilde{R}'}(d_1, \ldots, d_m)\}$$

$$\mu_{\tilde{R}\cap\tilde{R}'}(d_1, \ldots, d_m) = \min\{\mu_{\tilde{R}}(d_1, \ldots, d_m), \mu_{\tilde{R}'}(d_1, \ldots, d_m)\}$$

where $(d_1, \ldots, d_m) \in D_1 \times \cdots \times D_m$

The underlying principles in the above definition are the following. If an element shall be in two fuzzy relations (intersection), then we take the weakest link in the chain, i.e., we compute the minimum of the membership grades. However, if we have the choice between two relations (union), we make the best out of it and take the maximum of the membership grades. We admit that this is a sloppy justification for the above definition; please refer to the fuzzy literature for a more precise justification (e.g., [16]).

In the literature about fuzzy sets, you can also find alternative ways of defining the intersection and union of fuzzy sets. The different definitions are based on different set of axioms that are expected to hold for the intersection and union, and thus each of them is valuable from some certain point of view. In our opinion, the definitions we have chosen here are adequate for our purposes, as they are based on simple and intuitive principles, and they have the advantage of ensuring that we don't end up with indefinite small membership grades.

Beside union and intersection, two more operations are required: projection and cylindrical extension. The reason why these operations are useful for fuzzy constraint networks is that the relations of the constraints in the network may have different product spaces. In order to combine the relations, we must inject them into a bigger product space that subsumes all product spaces occurring in the network (cylindrical extension). To reverse this process, we must project from the all-subsuming product space on to the individual product spaces (projection). Formally, this can be expressed as follows:

Definition 7 (Projection)
Let $\tilde{R}$ be a fuzzy relation in the product space $D_1 \times \cdots \times D_m$, then the i-th projection of $\tilde{R}$, $i \in \{1, \ldots, m\}$, is defined as

$$\tilde{R}^{\downarrow i} = \{\langle d_i, \mu_{\tilde{R}^{\downarrow i}}(d_i)\rangle \mid d_i \in D_i\}$$

with

$$\mu_{\tilde{R}^{\downarrow i}}(d_i) = \max\{\mu_{\tilde{R}}(d'_1, \ldots, d'_m) \mid (d'_1, \ldots, d'_m) \in D_1 \times \cdots \times D_m \wedge d'_i = d_i\}$$

In general, the projection of $\tilde{R}$ on $D_{i_1} \times \cdots \times D_{i_k}$, $i_1, \ldots, i_k \in \{1, \ldots, m\}$, is defined as

$$\tilde{R}^{\downarrow i_1,\ldots,i_k} = \{\langle (d_{i_1}, \ldots, d_{i_k}), \mu_{\tilde{R}^{\downarrow i_1,\ldots,i_k}}(d_{i_1}, \ldots, d_{i_k})\rangle \mid (d_{i_1}, \ldots, d_{i_k}) \in D_{i_1} \times \cdots \times D_{i_k}\}$$

with

$$\mu_{\tilde{R}^{\downarrow i_1,\ldots,i_k}}(d_{i_1}, \ldots, d_{i_k}) = \max\{\mu_{\tilde{R}}(d'_1, \ldots, d'_m) \mid (d'_1, \ldots, d'_m) \in D_1 \times \cdots \times D_m \wedge \forall j \in \{i_1, \ldots, i_k\} : d'_j = d_j\}$$

Definition 8 (Cylindrical Extension)
Let $\tilde{R}$ be a fuzzy relation in the product space $D_{i_1} \times \cdots \times D_{i_k}$ with $\{D_{i_1}, \ldots, D_{i_k}\} \subseteq \{D_1, \ldots, D_m\}$, then the cylindrical extension $\tilde{R}^{\uparrow 1,\ldots,m}$ of $\tilde{R}$ in $D_1 \times \cdots \times D_m$ is the largest fuzzy relation in $D_1 \times \cdots \times D_m$ whose projection on $D_{i_1} \times \cdots \times D_{i_k}$ is $\tilde{R}$.

Definitions 6–8 provide us with the necessary means for solving a fuzzy CSP, i.e., determining the fuzzy relation $\tilde{R}$ represented by a fuzzy constraint network $\tilde{N}$. If $\tilde{N} = \langle\{\tilde{V}_1, \ldots, \tilde{V}_m\}, \{\tilde{C}_1, \ldots, \tilde{C}_n\}\rangle$, where $\tilde{V}_i$ is a fuzzy variable in some domain D_i for each

$i \in \{1, \ldots, m\}$, then $\tilde{R}$ is a fuzzy relation in the product space $D_1 \times \cdots \times D_m$ such that for each $\tilde{C}_j$, $j = 1, \ldots, n$, the following holds: Let $\tilde{V}_{i_1}, \ldots, \tilde{V}_{i_k}$ be the variables of $\tilde{C}_j$ and $\tilde{R}_j$ its relation, then $\tilde{R}^{\downarrow i_1, \ldots, i_k} = \tilde{R}_j$. From the more constructive viewpoint, $\tilde{R}$ can be defined as the intersection of relations $\tilde{R}'_j$, $j = 1, \ldots, n$, where $\tilde{R}'_j$ is the cylindrical extension of $\tilde{R}_j$ in $D_1 \times \cdots \times D_m$, i.e., $\tilde{R}'_j = \tilde{R}_j^{\uparrow 1, \ldots, m}$.

In most cases, we are not interested in an arbitrary element of the fuzzy relation given by the network, but want to obtain an element whose membership grade is beyond a certain threshold. (Such an element is called quasi-solution of the crisp constraint network from which the fuzzy constraint network was derived.) We therefore define:

Definition 9 ((Strong) α-Level Sets)
Let $\tilde{R}$ be a fuzzy relation in the product space $D_1 \times \cdots \times D_m$, then the (crisp) set of elements that belong to the fuzzy relation $\tilde{R}$ at least to the degree α is called the α-level relation of $\tilde{R}$. If the degree of the elements is greater than α, the set is called the strong α-level relation of $\tilde{R}$:

$$R_\alpha = \{d_1, \ldots, d_m \in D_1 \times \cdots \times D_m \mid \mu_{\tilde{R}}(d_1, \ldots, d_m) \geq \alpha\} \quad (\alpha\text{-level relation})$$
$$R_{\overline{\alpha}} = \{d_1, \ldots, d_m \in D_1 \times \cdots \times D_m \mid \mu_{\tilde{R}}(d_1, \ldots, d_m) > \alpha\} \quad (\text{strong } \alpha\text{-level relation})$$

Taking this definition as a basis, we define an α-solution of a given fuzzy CSP as follows:

Definition 10 ((Strong) α-Solution)
Let $\tilde{N} = \langle\{\tilde{V}_1, \ldots, \tilde{V}_m\}, \{\tilde{C}_1, \ldots, \tilde{C}_n\}\rangle$ be a fuzzy constraint network, where $\tilde{V}_i$ is a fuzzy variable in some domain D_i, $i = 1, \ldots, m$. Further, let $\tilde{R}$ be the relation given by $\tilde{N}$, i.e., the intersection of the cylindric extensions of the fuzzy relations given by $\tilde{C}_1, \ldots, \tilde{C}_n$. Then $(d_1, \ldots, d_m) \in D_1 \times \cdots \times D_m$ is

- *an α-solution of N if $(d_1, \ldots, d_m) \in R_\alpha$*
- *a strong α-solution of N if $(d_1, \ldots, d_m) \in R_{\overline{\alpha}}$*

Of course, knowing what an α-solution of a constraint network means is only part of the story. The other part is to implement an efficient algorithm for finding such a solution. Although this part depends on the underlying domains, there are some techniques that can be used in general to obtain such an algorithm. We will discuss these techniques in the next section.

4 Algorithms

There are two classes of algorithms for fuzzy constraint satisfaction:

1. Algorithms that find an α-solution for a given fuzzy constraint network if such a solution exists.

2. Algorithms that preprocess a given fuzzy constraint network and transform it into a network that is (hopefully) easier to solve.

The fuzzy constraint satisfaction algorithms of class 1 are NP-complete. Usually, their purpose is finding the best solution, i.e., a solution with maximal α. Since the maximal α is unknown in most cases, the task almost always turns into an optimization problem. For this type of constraint satisfaction problems, various approaches have been suggested in the constraint literature. An example is the Boltzmann machine described in [11]. It can be adopted to fuzzy constraint networks in a straightforward way.

The idea of [11] is to associate penalties with the elements of the constraint relations and to transform the resulting network into a Boltzmann machine. The penalties denote how acceptable the relation elements are. Penalties are usually integers greater than or equal to 0, where 0 is associated with optimal relation elements. In most cases, membership grade can easily be transformed into penalties as follows:

$$p = \lfloor \sigma(1 - \mu) \rfloor$$

In this formula, σ denotes a suitable stretching factor, usually some big integer.

The algorithms that are in class 2 are usually of polynomial time complexity. Examples of such algorithms are F-AC3 for computing arc consistency and F-PC2 for computing path consistency [3].

5 Conclusion

Real world problems (CSPs, in particular) are often inconsistent and therefore require some means of representing fuzziness. The approach introduced in this paper provides such means. As opposed to other approaches, we do not assign fuzzy values to the constraints of a given constraint network but to the elements of the constraint relation. A result is greater flexibility and applicability.

The testbed for our approach is the domain of spatiotemporal reasoning. There is evidence that, when reasoning about space and time, one has to deal with imprecise and uncertain spatial and temporal relations [6], and that fuzzy constraints are a plausible approach to this [10].

References

[1] C. Alsina, E. Trillas, and L. Valverde. On some logical connectives for fuzzy sets theory. In D. Dubois, H. Prade, and R.R. Yager, editors, *Readings in Fuzzy Sets for Intelligent Systems*, pages 71–76. Morgan Kaufmann, San Mateo, California, 1993.

[2] Y. Descotte and J.C. Latombe. Making compromises among antagonist constraints in a planner. *Artificial Intelligence*, 27:183–217, 1985.

[3] D. Dubois, H. Fargier, and H. Prade. Propagation and satisfaction of flexible constraints. Rapport IRIT/92-59-R, IRIT, Toulouse Cedex, France, 1992.

[4] J.C. Fodor and T. Keresztfalvi. Non-conventional conjunctions and implications in fuzzy logic. In *Fuzzy Logic in Artificial Intelligence: Proc. Austrian Artificial Intelligence Conference*, pages 16–26, Linz, Austria, 1993.

[5] B.N. Freeman-Benson, J. Maloney, and A. Borning. An incremental constraint solver. *Communications of the ACM*, 33:54–63, 1990.

[6] C. Freksa. Qualitative spatial reasoning. In *Proc. Workshop RAUM*, pages 21–36, Koblenz, Germany, 1990.

[7] E.C. Freuder. Partial constraint satisfaction. In *Proc. IJCAI-89*, pages 278–283, Detroit, Michigan, 1989.

[8] H.W. Guesgen. *CONSAT: A System for Constraint Satisfaction.* Research Notes in Artificial Intelligence. Morgan Kaufmann, San Mateo, California, 1989.

[9] H.W. Guesgen and J. Hertzberg. A constraint-based approach to spatiotemporal reasoning. *Applied Intelligence (Special Issue on Applications of Temporal Models)*, 3:71–90, 1993.

[10] H.W. Guesgen and J. Hertzberg. Spatial persistence: Preliminary report. In *Proc. IJCAI-93 Workshop on Spatial and Temporal Reasoning*, pages 11–34, Chambéry, France, 1993.

[11] J. Hertzberg and H.W. Guesgen. Transforming constraint relaxation networks into Boltzmann machines. In *Proc. GWAI-91*, pages 244–253, Bonn, Germany, 1991.

[12] J. Hertzberg, H.W. Guesgen, A. Voß, M. Fidelak, and H. Voß. Relaxing constraint networks to resolve inconsistencies. In *Proc. GWAI-88*, pages 61–65, Eringerfeld, Germany, 1988.

[13] W. Silvert. Symmetric summation: A class of operations on fuzzy sets. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(10):657–659, 1979.

[14] S. Weber. A general concept of fuzzy connectives, negations and implications based on t-norms and t-conorms. *Fuzzy Sets and Systems*, 11:115–134, 1983.

[15] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[16] Hans-Jürgen Zimmermann. *Fuzzy Set Theory and its Applications.* Kluwer-Nijhoff Publishing, Dordrecht, The Netherlands, 1985.