A Tractable Subclass of Fuzzy Constraint Networks

Alfonso Bosch[1], Francisco Guil[1], and Roque Marin[2]

[1] Dpto. de Lenguajes y Computación, Universidad de Almeria, 04120 Almeria (Spain)
{abosch, fguil}@ual.es
[2] Dpto. de Ingenieria de la Informacion y las Comunicaciones,
Universidad de Murcia, Campus de Espinardo, 30071 Espinardo (Murcia, Spain)
roque@dif.um.es

Abstract. The Fuzzy Constraint Networks model, a generalization of the Disjunctive Temporal Fuzzy Constraint Networks, is a framework that allows representing and reasoning with fuzzy qualitative and quantitative complex constraints. However, its general complexity is exponential, and we need to find tractable subclasses. In this paper we propose two algorithms to deal with a tractable subclass named Series-Parallel Fuzzy Constraint Networks.

1 Introduction

Fuzzy Temporal Constraint Networks model (FTCN), introduced in [6], allows expressing simple constraints, representing them by means of a convex and normalized possibility distribution. Fuzzy temporal constraints allow combining precise, imprecise, qualitative and quantitative information. Then, this model is suitable for temporal reasoning in domains where the combination of such constraint types is required. A fuzzy model allows intermediate consistency degrees, and to quantify the possibility and necessity of a relationship or query. In addition, constraint propagation reduces one of the drawbacks associated with fuzzy reasoning, the degradation of distributions when chaining fuzzy rules (reduction of the core, and enlargement of the support of the possibility distributions).

In certain tasks, such as planning or scheduling, a more general model with disjunctions is needed. Then, the FTCN model is enhanced, defining a constraint with a finite set of possibility distributions, normalized and convex, obtaining the Disjunctive Fuzzy Temporal Constraint Networks (DFTCN) [1]. This model extends the TSCP framework proposed by Dechter [3], allowing constraints such as "Irrigation is much before or a little after than Treatment", and subsumes the Vilain & Kautz Point Algebra [9]. This framework allows representing all the possible relationships between time points, between intervals and between time points and intervals, and their disjunctions.

The main drawback of DFTCN is its computational inefficiency, because generally these networks are non-decomposable, needing backtracking to find a solution [1]. Determining the consistency and computing the minimal network are also exponential. With small problems, this is not a drawback, but in order to generalize the use of the model in a general scope, it would be interesting to find tractable subclasses, as series-parallel networks [2].

R. Moreno Díaz et al. (Eds.): EUROCAST 2005, LNCS 3643, pp. 269 – 274, 2005.
© Springer-Verlag Berlin Heidelberg 2005

In this paper we use a generalization of DFTCN, named Fuzzy Constraint Networks (FCN) [2], and we present two efficient algorithms, SP^+ and SPR, to decide if a FCN is consistent, series-parallel and obtain its equivalent path-consistent (and minimal) network, using the information of the original problem constraints.

The remainder of this paper is organized as follows. Section 2 describes the FCN model; Section 3 describes a tractable subclass named Series-Parallel Networks; Section 4 presents two efficient algorithms to deal with this Series-Parallel Networks and Section 5 summarizes the conclusions and presents the future work.

2 Fuzzy Constraint Networks

A Fuzzy Constraint Network (FCN) L^d consists of a finite set of n+1 variables $X_0, \ldots,$ X_n (X_0 as origin for problem variables), whose domain is the set of real numbers $\boldsymbol{R}$, and a finite set of disjunctive binary constraints L^d_{ij} among these variables. X_0 is a variable added to use only binary constraints, and it can be assigned to an arbitrary value (for simplicity's sake, this value is usually 0).

A disjunctive binary constraint L^d_{ij} among variables X_i, X_j is defined with a finite set of possibility distributions, $\{\pi^1_{ij}, \pi^2_{ij}, \ldots, \pi^k_{ij}\}$ normalized and convex [5], defined over the set of real numbers $\boldsymbol{R}$; for $x \in \boldsymbol{R}$, $\pi_m(x) \in [0,1]$ represents the possibility that a quantity m can be precisely x.

A value assignation for variables X_i, X_j, $X_i=a$; $X_j=b$, a, $b \in \boldsymbol{R}$, satisfies the constraint L^d_{ij} iff it satisfies one of its individual constraints:

$$\exists \pi^p_{ij} \in L^d_{ij} \,/\, \pi^p_{ij}(b-a) > 0 \tag{1}$$

The maximum possibility degree of satisfaction of a constraint L^d_{ij} for an assignment $X_i = a$, $X_j = b$ is

$$\sigma^{\max}_{ij}(a,b) = \max_{1 \le p \le k} \pi^p_{ij}(b-a) \tag{2}$$

A constraint L^d_{ij} among variables X_i, X_j defines a symmetric constraint L^d_{ji} among X_j, X_i, and the lack of a constraint is equivalent to the universal constraint π_U. A FCN can be represented with a directed graph, where each node corresponds to a variable and each arc corresponds to a constraint between the connected variables, omitting symmetric and universal constraints. The set of possible solutions of a FCN L^d is defined as the fuzzy subset from $\boldsymbol{R}^n$ associated to the possibility distribution given as:

$$\pi_S(v_1, \ldots, v_n) = \min_{\substack{0 \le i \le n \\ 0 \le j \le n}} (\sigma^{\max}_{ij}(v_i, v_j)) \tag{3}$$

An n-tuple $V = (v_1, \ldots v_n) \in \boldsymbol{R}^n$ of precise values is an σ-possible solution of a FCN L^d if $\pi_S(V) = \sigma$. We say that a FCN L^d is consistent if it is 1-consistent, and it is inconsistent if it does not have any solution.

Given a FCN L^d, it is possible to find out several networks which are equivalent to L^d. We can obtain this networks using the composition and intersection operations, defined in [1] for temporal reasoning. Among all the equivalent networks, there is always a network M^d that is minimal. This network contains the minimal constraints. If

M^d contains an empty constraint, L^d is inconsistent. If p if the maximum of possibility distributions in each constraint, and the network has q disjunctive constrains and n variables, then the minimal network M^d of a FCN L^d can be obtained with a complexity $O(p^q n^3)$, where n^3 is the cost of solving each case non disjunctive FCN [7]. Due to this exponential complexity, we need to find a more practical approach.

3 Series-Parallel Fuzzy Constraint Networks

A network is series-parallel [8] in respect to a pair of nodes *i,j* if it can be reduced to arc (*i,j*) applying iteratively this reduction operation: a) select a node with a degree of two or less; b) remove it from the network; c) connect its neighbours. A network is series-parallel if it is series-parallel in respect to every pair of nodes.

The basic algorithm for checking if a network is series-parallel has an $O(n^3)$ complexity, and there is a more efficient algorithm that checks this property with an $O(n)$ complexity [10], applied to fault-tolerant networks (IFI networks). In [2], we introduced the SP algorithm, a variant of the later approach for constraint networks.

Series-parallel networks present some interesting properties. If a network is series-parallel, the path-consistent network is the minimal network, although the intersection and composition operations are non-distributive [8]. As a subproduct of checking whether a network is series-parallel, a variable ordering is obtained when deleting the nodes. Applying directional path-consistency (DPC) algorithm [4] in the reverse order, a backtrack-free network is obtained and the minimal constraint between the first two variables of the ordering too. This can be interesting when we need only to compute a minimal constraint for two variables, and not the minimal network. In addition, if the network is series-parallel, we can decide absolutely whether the network is consistent, by applying DPC algorithm in the reverse order. Using all these properties, we can say that series-parallel Fuzzy Constraint Networks are a tractable subclass of Fuzzy Constraint Networks, because we can manage it without backtracking.

4 Algorithms for Series-Parallel Fuzzy Constraint Networks

The SP algorithm decides whether a network is series-parallel using the graph topology. After the SP test, we need to use another algorithms to manage the network. But if we enhance SP algorithm to use constraint information, we can process the constraints, deciding simultaneously whether the network is series-parallel or not.

Our initial proposal is to include a constraint relaxation operation when a node or variable of degree two is eliminated. This operation updates a constraint, deleting the values disallowed by a path of length two, ensuring that the constraint will be path-consistent in respect to this path.

Working in this direction, we propose the algorithm SP^+, which can decide with an $O(n)$ complexity whether a network is inconsistent if it detects an empty constraint. If an empty constraint is not detected and the algorithm decides that the network is series-parallel, the network is consistent. As a subproduct, the final obtained constraint is the minimal constraint among the two lasting variables, and the equivalent network obtained with the computed constraints is decomposable in the reverse variable reduction ordering. This is the code of SP^+ algorithm:

SP⁺ Algorithm

```
begin
  for each i=0..n Compute-degree (i)
  NodeQueue = {nodes with degree 1 and 2}
  while (NodeQueue <> ∅ and |V| > 2)
    begin
    node=Get(NodeQueue)
      if Degree(node) = 2 then     L_ij ← L_ij ∩̇ L_ik ⊕̇ L_kj
      if L_ij = ∅ then exit
        V ← V − {node}
        if Degree(node)=1 then Degree(Neighbour(node)) --
        if Degree(Neighbour(node)) = 2 then
              Put(NodeQueue,Neighbour(node))
            else if Connected(Neighbours(node)) then
              Degree(Neighbour(node)) --
        if Degree(Neighbours(node)) = 2 then
              Put(NodeQueue, Neighbours(node))
            else E ← E + {ArcNeighbours(node)}
   end
 if (NodeQueue = ∅ and |V| > 2) then
             exit ("Network is not series-parallel")
end
```

Next, we will enumerate the properties of the algorithm.

Lemma 1.- If the series-parallel reduction algorithm stops because it computes an empty constraint, the network is inconsistent.

Lemma 2.- If the series-parallel reduction algorithm stops without detecting an empty constraint, and it decides that the network is series-parallel, then the network is consistent.

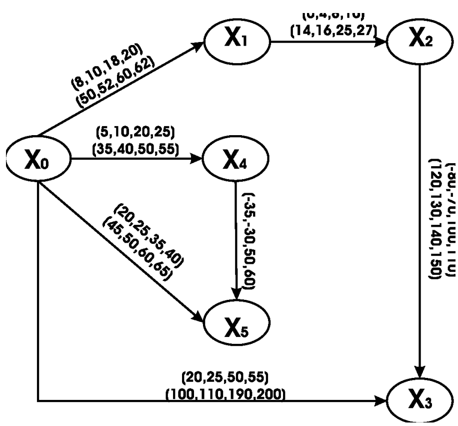

Fig. 1. Example of network to be reduced with SP⁺

Lemma 3.- If series-parallel reduction algorithm terminates without detecting an empty constraint and it decides that the network is series-parallel, then the final constraint obtained is the minimal constraint among the two lasting variables.

Figure 1 shows a FCN network to be reduced with SP+. Figure 2 shows the reduction process and refined constraints obtained.

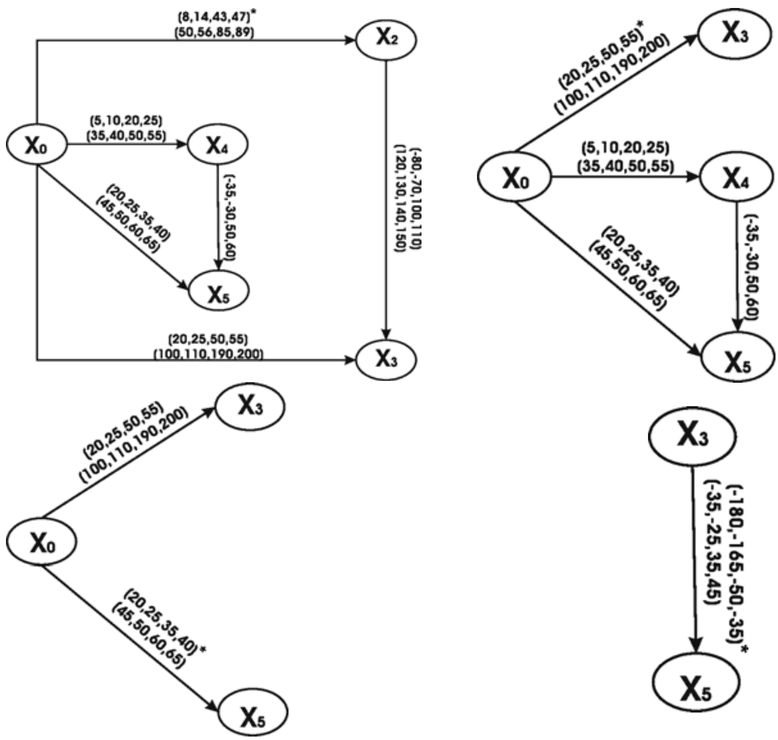

Fig. 2. Example of reduction process

Based on the results of SP+, we can propose a linear rebuilding algorithm SPR, which obtains a path-consistent network respecting the original network topology. The obtained network is not the full path-consistent or minimal network, because it is not a complete graph.

5 Conclusions and Future Work

In this paper, we have defined Series-Parallel Fuzzy Constraint Networks as a tractable subclass of Fuzzy Constraint Networks. We have presented the SP+ linear algorithm for checking simultaneously if a constraint network is series-parallel and it has a solution, obtaining a minimal constraint among two variables too.

Using the results of SP^+, we can write a linear rebuiding algorithm for obtaining a path-consistent network, with the original network topology.

As future work, we can highlight the evaluation of these techniques and its comparation versus other approaches for series-parallel networks, the implementation of a linear algorithm for obtaining the minimal network, and the search of another FCN tractable subclasses and decomposition methods.

References

1. A. Bosch, M. Torres, and R. Marín. Reasoning with Disjunctive Fuzzy Temporal Constraint Networks. In *Proc. of 9th Int. Symposium on Temporal Representation and Reasoning (TIME'02)*. IEEE Computer Society Press, pages 36-43, 2002.
2. A. Bosch, F. Guil, C. Martinez, and R. Marín. Series-Parallel and Tree-Decomposition Approaches for Fuzzy Constraint Networks. In *Proc. of the 8th Ibero-American Conference on Artificial Intelligence (IBERAMIA'02)*. LNAI, 2527:275-284, 2002.
3. R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. *Artificial Intelligence*, 49:61-95, 1991.
4. R. Dechter, and J. Pearl, "Network-based heuristics for constraint-satisfaction problems", *Artificial Intelligence*, 34, Elsevier, 1987, pp. 1-38
5. D. Dubois, H. Prade, *Possibility Theory: An approach to computerized processing of uncertainty*, Plenum Press, New York, 1988.
6. R. Marin, S. Barro, A. Bosch, and J. Mira. Modeling time representation from a fuzzy perspective. *Cybernetic & Systems*, 25(2):207-215, 1994.
7. R. Marín, M. Cardenas, M. Balsa, and J. Sanchez, "Obtaining solutions in fuzzy constraint networks", *Int. Journal of Approximate Reasoning*, 16, Elsevier, 1997, pp. 261-288.
8. U. Montanari, "Networks of constraints: fundamental properties and applications to picture processing", *Information Science*, 7, 1974, pp. 95-132.
9. E. Schwalb, and L. Vila, "Temporal Constraints: A Survey", *Constraints* 3 (2/3), 1998, pp. 129-149.
10. J.A. Wald, and C.J. Colburn, "Steiner Trees, Partial 2-Trees and Minimum IFI Networks", *Networks,* 13, 1983, pp. 159-167.