2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology

On Solving Distributed Fuzzy Constraint Satisfaction Problems with Agents

Xuan Thang Nguyen and Ryszard Kowalczyk
Centre for Information Technology Research
Swinburne University of Technology
Melbourne VIC 3122, Australia
{xnguyen, rkowalczyk}@ict.swin.edu.au

Abstract

Fuzzy Constraint Satisfaction Problem (FCSP) offers flexible modelling and solving of many real world problems such as scheduling and resource allocation. In this paper, we focus on Distributed Fuzzy Constraint Satisfaction problems (DisFCSP) where fuzzy constraints and variables of the problems are distributed among a set of collaborative agents. We propose two approaches to solve these problems: An iterative method and an adaptation of the Asynchronous Distributed constraint OPTimization algorithm (ADOPT) for solving DisFCSP. We also present experiments on the performance comparison of the two approaches.

1 Introduction

Distributed Constraint Satisfaction (DisCSP [1]) has been one of the most active research area of the Multi-Agent-System and and Aritifical Intelligence communities in the last few years. Genarally, a DisCSP is a problem with finite number of variables, each of which has a finite and discrete set of possible values and a set of constraints over the variables. These variables and constraints are distributed among a set of autonomous and communicating agents. These agents in general are collaborative agents. A solution in DisCSP is an instantiation of all variables such that all the constraints are satisfied. Distributed negotiation is an example of DisCSP where different agents share some common objectives but also abide to different constraints and have their own preferences.

Whilst a number of approaches have been proposed to solve DisCSPs [6, 5] or centralized FCSP [4] alone, only a few work is related to the combination of DisCSP and FCSP. This paper focuses on Distributed Fuzzy Constraint Satisfaction problems (DisFCSP) which extends both DisCSP and FCSP. A DisFCSP can be considered as either a version of a DisCSP where every constraint is fuzzy, or an extension of a FCSP in distributed environments. An example of DisCSPs is Multi-Agent System (MAS) negotiation where the agents' preferences over values of negotiation attributes are modelled as fuzzy sets.

This paper is organized as follows. A literature review on DisFCSP is presented in the next section. In Section 3, we give an overview of important concepts used in DisCSP. In Section 4, we present the FABT algorithm which is based on the Asynchronous Backtracking (ABT) algorithm [6] to solve the DisFCSP. Section 5 presents our experiments with a discussion. Finally, conclusions and future work are discussed in Section 6.

2 Related Work

DisFCSP has been of interest to the MAS community, especially in the context of distributed resource allocation, collaborative scheduling, and negotiation (e.g. [3]). Those works focus on bilateral negotiations and when many agents take part, a central coordinating agent may be required. For example, the work in [3] promotes a rotating coordinating agent which acts as a central point to evaluate different proposals sent by other agents. Hence the network model employed in those work is not totally distributed. Another important note is that these work focuses on competitive negotiation where agents try to outsmart each other as opposed to collaborative negotiation hence does not use techniques from DisCSP algorithms.

Another related work is the iterative approach presented in [2] for hierachical DisCSPs (and may also be used for DisFCSPs). Our interative method is based on that approach. However, the iteration in the approach is either from top down or bottom up. Therefore the complexity in the worst case is $O(n)$ (calls to DisCSP instances) whereas our new iterative based approach in this paper is $O(log_2 n)$

[1]Some references use DCSP for Distributed Constraint Satisfaction. Here we use DisCSP to differentiate from Dynamic Constrain Satisfaction which are also referred as DCSP in CSP literature

0-7695-3027-3/07 $25.00 © 2007 IEEE
DOI 10.1109/IAT.2007.81

where n is the number of α-cut levels as will be explained in Section 4.1.

3 The ABT4F algorithm

This part describes the ABT based algorithm for Fuzzy constraints (ABT4F). The algorithm enables the iterative principle [4] to be used efficiently in DisFCSP by adding two new features: a built-in termination mechanism and a nogood caching technique for reusing information between successive searches

It is well known that a fuzzy set can be viewed as a collection of crisp sets at different α-cut levels (i.e. the principle of the resolution identity). Similarly, a Fuzzy CSP can be modelled as a collection of crisp CSPs at different levels of constraint satisfaction:

$$c_j(x) = \sum \alpha c_j^{\alpha}(x) \tag{1}$$

in which c_j^{α}(x) is a crisp constraint which is satisfied iff x belongs to the α-cut: $\{x : \mu_{c_j}(x) \geq \alpha, 0 \leq \alpha \leq 1\}$.

The iterative methods [4] for centralized search use this model to solve a FCSP as a series of CSPs. Using the same principle, a DisFCSP $\mathbf{P}^f$ in general can be considered as a union of Dis(Crisp)CSPs at different levels of constraint satisfactions:
$\mathbf{P}^f = \sum \alpha \mathbf{P}_{\alpha}$ where a P_{α} results from $\mathbf{P}^f$ by replacing all fuzzy constraints in $\mathbf{P}^f$ with their crisp constraints at the α level. A solution of the DisFCSP in definition 2 is to find a solution at the highest α-cut level.

The principle of an iterative approach is to select and solve a series $\{\mathbf{P}_{\alpha_1}, ..., \mathbf{P}_{\alpha_m}\}$ of DisCSP instances within DisFCSP. The decision for selecting $\mathbf{P}_{\alpha_{i+1}}$ partly depends on the outcome of the search at $\mathbf{P}_{\alpha_{i+1}}$. In particular:

$$\alpha_{i+1} \geq \alpha_{LB} = max\,\{\alpha_k : k \leq i, P_{\alpha_k}\ solved\} \tag{2}$$
$$\alpha_{i+1} \leq \alpha_{UB} = min\,\{\alpha_k : k \leq i, P_{\alpha_k}\ overconstrained\} \tag{3}$$

Two simple methods of selecting the next α-cut level are to use a top-down or bottom-up approach. In our algorithm, we use a strategy similar to the binary search for this selection. The main idea is to continuously update a global α_{LB} (lower bound) and a α_{UB} (upper bound). The α_{LB} is the satisfaction level at which we know that a solution can be found. The α_{UB} is the satisfaction level above that we cannot find any solution for. The search level is computed as the middle point between the α_{LB} and α_{UB}:

$$\alpha_i = (\alpha_{LB} + \alpha_{UB})/2 \tag{4}$$

With this binary search based selection strategy, in the worst case for random constraints, we make $\log_2 \ell$ Dis(crisp)CSP search as opposed to ℓ search in top-down or bottom-up approaches where ℓ is the total number of different satisfaction levels. At each α-cut level, FABT uses a ABT algorithm. In addition to ABT message types (see [6]), FABT uses another message type called *change-threshold* which is used by an agent in order to instruct every other agent to change to a new DisCSP problem.

4 ADOPT adaption for DisFCSP

We present here a brief summary of the ADOPT algorithm. For more information on ADOPT, we refer to [5]. In ADOPT, a number of n agents collaborate to solve the optimization problem:

$$min \sum_{1 \leq i \leq j \leq n} f_{ij}(v_i, v_j) \tag{5}$$

in which for every different pair i, j $f_{ij}(d_i, d_j)$ is a function which defines the cost incurred when agents A$_i$ and A$_j$ select the values d$_i$ and d$_j$ for their variables v$_i$ and v$_j$ respectively.

During the search, an agent A$_i$ (i.e. the agent holding v$_i$) continues to update its LB and UB. These values when A$_i$ selects the assignment $v_i = d_i$ is computed as:

$$\forall d_i \in D_i, LB(d_i) = \delta(d_i) + \sum_{A_l \in Children} lb(d_i, v_l) \tag{6}$$
$$\forall d_i \in D_i, UB(d_i) = \delta(d_i) + \sum_{A_l \in Children} lb(d_i, v_l) \tag{7}$$

whereas $lb(d_i, v_l)$ the is LB reported previously from A$_l$- a children of A$_i$ for A$_i$' assignment of v_i=d_i. $\delta(d_i)$ is the local cost incurred by shared constraints between A$_i$ and its parents.

$$\delta(d_i) = \sum_{(v_j, d_j) \in View} f_{ij}(d_i, d_j) \tag{8}$$

In general, A$_i$ tries to select an assignment of its own value so that LB or UB is smallest (to the final aim of making the cost smallest). In other words, the LB and UB choosen in general is:

$$LB = min_{d_i \in D_i} LB(d_i) \tag{9}$$
$$UB = min_{d_i \in D_i} UB(d_i) \tag{10}$$

ADOPT adaptation for DisFCSP consists of two steps. The first step is to compute ADOPT cost from the fuzzy constraints at each agent in a DisFCSP. To do this, we can define the cost for an assignment of v_i at the agent A$_k$ as:

$$f_{ij}(v_i, v_j) = \begin{cases} 1 - sat_{A_i}(v_i) & \text{if i=j} \\ 0 & \text{if i}\neq\text{j} \end{cases}$$

where sat_{A_i} is defined in (3). The global objective of the DisFCSP is:

$max \quad min_{1 \leq i \leq n} \{sat_{A_i}(v_i)\} \Leftrightarrow min \quad max_{1 \leq i \leq n} \{f_{ij}(v_i, v_j)\}$ for ADOPT. Note that we have a maximization for the aggregator instead of a summation in (10).

The second step for ADOPT adaptation for DisFCSP is to redefine the computation of LB and UB since our global objective function is a *min max* operation. Instead of (11) and (12), we redefine these as:

$$\forall d_i \in D_i, LB(d_i) = max\{\delta(d_i), max_{\forall: A_l \in Children} lb(d_i, v_l)\} \quad (11)$$
$$\forall d_i \in D_i, UB(d_i) = max\{\delta(d_i), max_{\forall: A_l \in Children} lb(d_i, v_l)\} \quad (12)$$

For the changes above in the objective function and formulas of LB and UB, the same arguments in [5] for ADOPT's termination and correctness can still be applied. In other words, our ADOPT adaptation can be used to solve the DisFCSP problem.

5 Experiments

We carried out experiements for the two approaches on the distributed Map Colouring problem. In our problem, there are five colours (i.e. the domain size of the color variable is 5) of which each agent has different preference levels in $\{0.00, 0.25, 0.50, 0.75, 1.0\}$ These preferences are private to the agents. The agents collaborate to find the colours so that two neighboring agents use different colour while maximizing the minimum preference of agents (i.e. the max min problem).

We ran the experiments on different numbers of agents: from 8 to 50 agents on a Java discrete event simulation toolkit. The experimental implementation is available at [1]. On the average, the time taken for each synchronous cycle [6] is 25 seconds (simulation time). Different random map coloring instances were generated per run. 50 runs were used for each density value of 1.5, 2.0, and 3.0 (density = num of links/number of agents). Figure 1,2, and 3 in the Appendix plot the performance of FABT versus ADOPT in terms of completion time and traffic load (in total number of message per synchronous cycle). As can be seen from these figures, FABT outperforms ADOPT for hard problems (i.e. high density problems). In particular, for the density of 2.0, FABT provides better performance starting from 30 agents. For the density of 3 where many instances generated have a solution at zero level of preference (i.e overconstrained), FABT completely outperforms the ADOPT based algorithm. This can be explained by the *binary search* strategy that FABT uses. For high density problems FABT quickly proceed to check the lowest $alpha$-cut level (i.e. at 0.25) and as long as it discovers the problems are overconstrained, the algorithm is terminated. On the contrary, this early detection does not occur to the ADOPT based algorithm. It is important to know that this outperformance of FABT also comes with the cost of complexity in its nogood storage (as for ABT) whereas ADOPT maintains a linear complexity.

6 Conclusions

In this paper, we have proposed two approaches to solve DisFCSPs. In the first approach, we show how iterative techniques in centralized FCSP can be extended and used in distributed environments. This is explained with our FABT algorithm which solves DisFCSPs using successive ABT and provides termination mechanism as well as information reuse to ABT. In the second approach, we propose an adaption of the ADOPT algorithm. We carried out experiments to compare the performance of these two algorithms and draw the conclusion that ADOPT is more suitable for low density problems whereas FABT can have better performance for hard DisFCSP instances. However the performance of FABT come with the cost of complexity in its nogood storage. In summary, we believe that the proposed algorithms expand the applicability of FCSP to distributed problems that require distributed solving approaches such as resource allocation, collaborative scheduling and distributed negotiation.

References

[1] *SIM MAS Toolkit*. http://ciamas.it.swin.edu.au/software/simulator-doc/, 2007.

[2] K. Hirayama and M. Yokoo. An approach to overconstrained distributed constraint satisfaction problems: Distributed hierarchical constraint satisfaction, 2000.

[3] X. Luo, N. Jennings, N. Shadbolt, H. Leung, and J. Lee. A fuzzy constraint based model for bilateral multi-issue negotiations in semi-competitive environments.

[4] P. Meseguer and J. Larrosa. Computational complexity of fuzzy constraint satisfaction. In *Technical Report*, 1994.

[5] P. Modi, W. Shen, M. Tambe, and M. Yokoo. Adopt: Asynchronous distributed constraint optimization with quality guarantees, 2005.

[6] M. Yokoo and K. Hirayama. Algorithms for distributed constraint satisfaction: A review. *Autonomous Agents and Multi-Agent Systems*, 3(2):185–207, 2000.

7 Appendix

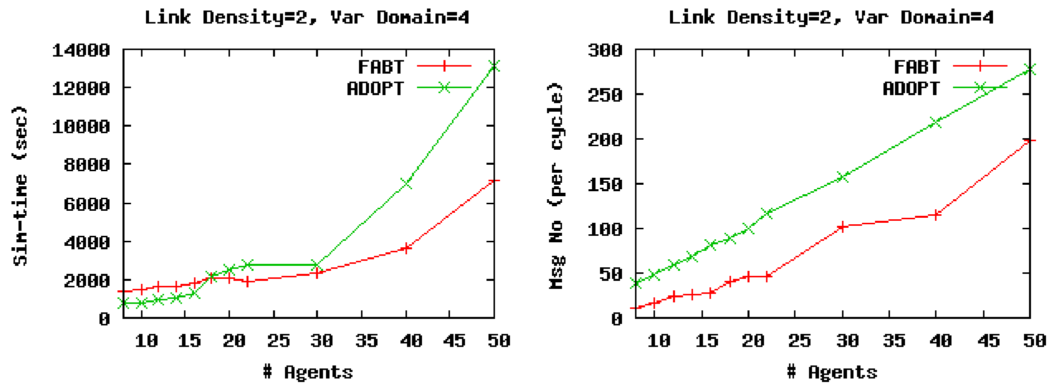

Figure 1. Comparation of FABT and ADOPT performance in terms of messages per cycle and completion time for density = 2.0

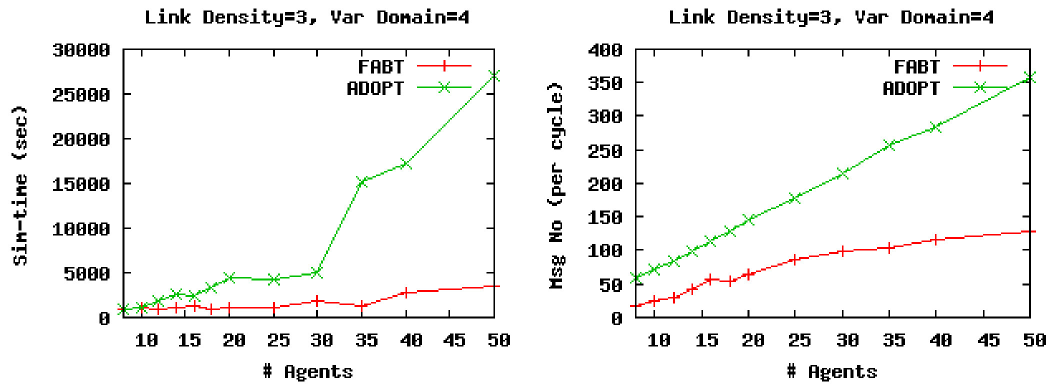

Figure 2. Comparation of FABT and ADOPT performance in term of message per cycle and completion time for density = 3.0

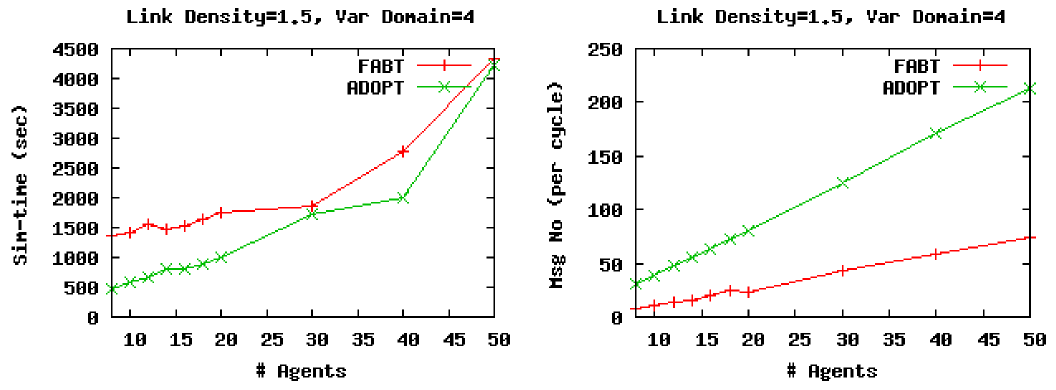

Figure 3. Comparation of FABT and ADOPT performance in term of message per cycle and completion time for density = 3.0