Distributed Constraint Satisfaction in a Wireless Sensor Tracking System

Ramon Bejar, Bhaskar Krishnamachari, Carla Gomes, and Bart Selman
Intelligent Information Systems Institute (IISI),
Department of Computer Science,
Cornell University, Ithaca, NY 14853
{bejar,bhaskar,gomes,selman}@cs.cornell.edu

Abstract

This paper describes our ongoing work on an interesting distributed constraint satisfaction problem (DCSP), SensorCSP, that is based on a system of wireless sensors tracking multiple mobile nodes. We present some preliminary results showing that the source of combinatorial complexity in this problem is closely linked to the level of communication in the system. This DCSP lends itself naturally to two models - one in which agents are associated with the sensors, and one in which agents are associated with the mobile nodes. We show that these models are *duals* of each other, and discuss how they differ in the number of intra and inter-agent constraints and how this might affect the cost of finding a distributed solution. We also suggest that a careful distinction must be made between explicit and implicit inter-agent constraints in this problem domain as this might affect the communication costs and the scalability of a distributed solution.

1 Introduction

With some of the recent developments in multi-hop wireless communication networks, distributed artificial intelligence techniques and approaches are increasingly seen as a useful tool. One of the areas in DAI is the use of distributed agents for cooperative problem solving through negotiation processes. If the common task of the agents can be described as one of choosing consistent values for certain variables while satisfying given constraints, a useful formalism to use in these contexts is that of a Distributed Constraint Satisfaction Problem (DCSP) [2, 13]. In a DCSP, there are assumed to n independent agents, each of which have some variables to which they need to assign values. There are, however, constraints between the variables for a single agent (intra-agent constraints) as well as constraints between the variables of different agents (inter-agent constraints) that must be satisfied.

We describe in this paper a problem domain we refer to as SensorCSP, an interesting test-bed for studying distributed constraint satisfaction. SensorCSP arises in the context of a distributed wireless sensor system for tracking mobile nodes. It is based on a challenge problem pertaining to the agent negotiating teams project (ANTs) [8].

Some of the current literature has focused on using DCSP to parallelize the solution of traditional constraint satisfaction problems such as SAT, Graph Coloring, N-Queens [5, 11, 13]. One property of the SensorCSP is that it lends itself readily to solutions involving the use of distributed agents, since the information needed to solve the problem is not localized.

We show in this paper that the origins of computational complexity in SensorCSP are directly related to the communication constraints of the system. Another interesting property of this domain is that when the distributed agents exchange information in order to solve the DCSP their interactions are restricted by these same constraints.

We present two different DCSP formulations of the SensorCSP that can be viewed as *duals* of each other. One of these representations has more inter-agent constraints and less intra-agent constraints than the other. One of goals in this work is to study how the choice of representation affects the complexity of computation and communication .

Some sound and complete algorithms have been developed for DCSP [4, 13]. These have proved useful in characterizing the computation and communication complexity of such problems, for example to examine phase transition behavior [5]. We make a distinction between explicit and implicit inter-agent constraints in SensorCSP. We show that in the SensorCSP domain, the use of a complete DCSP algorithm can sometimes entail indirect, multi-hop, communication between agents that have implicit inter-agent constraints.

The rest of this paper is organized as follows. In section 2 we describe the wireless sensor tracking system and define SensorCSP using a graph-based model. We prove that SensorCSP is an NP-complete problem using a reduction from the problem of partitioning a graph into isomorphic subgraphs [6]. We show that SensorCSP also exhibits a phase-transition phenomenon that is linked to the communication properties of the system. In section 3, we present the dual representations of this problem as a distributed constraint satisfaction problem. We then describe the communication issues that need to be considered when using asynchronous backtracking algorithms for solving this problem in section 4. Finally, we conclude with a discussion of our ongoing and future work in section 5.

2 The Wireless Sensor Tracking System

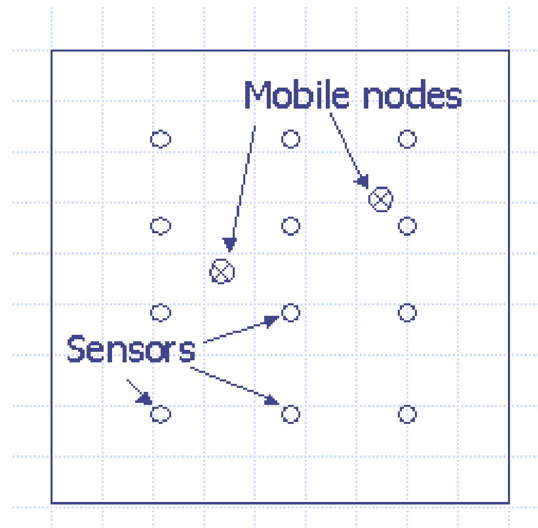

Figure 1: Sensors nodes and mobile nodes in an area

The wireless sensor tracking system consists of a set of n doppler radar-based sensors $S = \{s_1, s_2, \ldots s_n\}$ that are required to track the position of a set of m moving mobile nodes $T = \{t_1, t_2, \ldots t_m\}$. The radar coverage of each sensor is assumed to be a circular region of a fixed radius R. In order to estimate the location of each mobile node, the position and velocity measurements obtained from k different sensors must be combined. Thus, in order to be tracked, a mobile node must lie in the intersection of the coverage areas of at least k sensors. Figure 1 shows an instantaneous snapshot of such a system. Figure 2 shows the coverage zone for each mobile node. An edge is drawn between each sensor and all of its in-range mobile nodes.

We further assume that even if several mobile nodes are within range, each sensor can only be involved in tracking one of these at a given time. Each sensor node is equipped with sufficient computational resources. One of the requirements of the problem is that the tracking should be performed in a decentral-

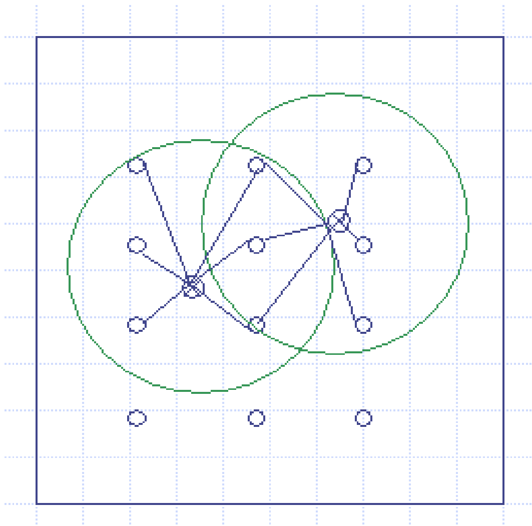

Figure 2: Coverage zone for each mobile node

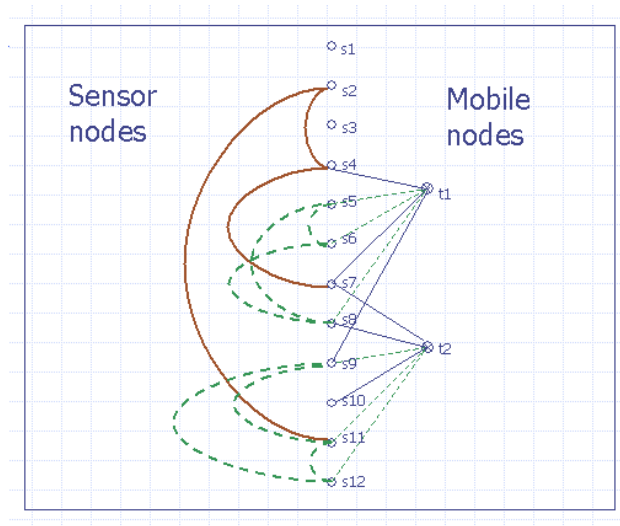

Figure 3: An example Graph G^* with communication and visibility edges between the sensor and mobile nodes: dashed edges represent a solution to Sensor-CSP in this graph

ized, distributed, manner.

Due to limited bandwidth and power resources, there may be time-varying constraints on communication between the nodes, i.e. at any given time direct communication may only be possible between some pairs of nodes. Only sensor nodes that can directly communicate with each other may track a particular mobile node.

2.1 Constraint Satisfaction Model

We can formulate the operation of the overall system, at any given moment in time, as a constraint satisfaction problem based on a graph model. We construct a graph $G^* = (S \cup T, E^*)$ where the vertices correspond to each of the sensors and mobile nodes at a particular instant in time. We place an edge between a sensor and a mobile node if the mobile node is within a distance R of the sensor. As we noted above, in the operational setting of the wireless sensor network, there may be constraints on the inter-sensor communication. To incorporate these into our graph model, we also place an edge $e' = (s_1, s_2)$ between all pairs of sensor nodes s_1 and s_2 that can communicate with each other. Thus, in addition to the visibility constraints modelled by the edges between sensors and mobile nodes, we have communication edges between sensor nodes.

Definition 1 *We say a node $t \in T$ is $k-$trackable, $k \in \mathbb{Z}^+$ by S in a graph $G^+ = (T \cup S, E^+)$ if the following hold:*

1. *There are exactly k edges touching node t, (k sensors are tracking this mobile node)*
2. *For all $s \in S$ such that $(s,t) \in E^+$, there is no other node $t' \in T$ for which $(s,t') \in E^+$ (each such sensor should track only this mobile node)*
3. *For all $s_1, s_2 \in S$ such that $(s_1,t) \in E^+$ and $(s_2,t) \in E^+$, $(s_1,s_2) \in E^+$ (all sensors tracking this mobile node can communicate with each other, pairwise).*

We formulate the constraint satisfaction problem in the following way:

SensorCSP: Given a graph $G^* = (S \cup T, E^*)$, is there a subgraph $G^+ = (S \cup T, E^+)$ of G^* such that every node $t \in T$ is $k-$trackable for some $k \in \mathbb{Z}^+$?

Claim 1 *SensorCSP is NP-complete.*

Proof: It is clear that it is in NP - given a certificate consisting of a graph G^+, it is easy to verify in polynomial time that it is indeed a subgraph of G^* and that all nodes $t \in T$ are $k - trackable$ by simply checking each of the conditions given in definition 1.

To complete the proof, we reduce the problem of partitioning a graph into isomorphic subgraphs to SensorCSP. This problem considers a graph $G = (V, E)$ and a graph $H = (V', E')$ such that $|V| = q \cdot |V'|$ for some $q \in \mathbb{Z}^+$. The question is to find a decomposition of the graph G into q disjoint subgraphs, such that every one of these graphs is isomorphic to H. This problem is NP-complete for every graph H that contains a connected component with at least three vertices [6]. Consider the particular sub-problem of finding a decomposition of G when H is a clique of size k and when $|V| = q \cdot k$ for some $q \in \mathbb{Z}^+$. We construct a new graph $G_{TS} = (V_{TS}, E_{TS})$ that is identical to $G = (V, E)$ except that we put q extra vertices W and we connect each of them to all the vertices in V. The set of vertices V represents the set of sensors and the edges E between them are communication edges. The additional set W of q vertices represent the set of mobile nodes of the G_{TS} graph. We need to show that the graph G has a decomposition into cliques of size k if and only if the graph G_{TS} has a subgraph such that every node $w \in W$ is $k - trackable$. Let us consider each direction of this equivalence separately. If the graph G has a decomposition into cliques of size k, then there must be q such cliques. In G_{TS}, each of these q disjoint cliques correspond to k communicating sensors. We can therefore assign each mobile node to such a distinct set of k communicating sensors. This is all that is required for G_{TS} to have a subgraph such that every node $w \in W$ is $k - trackable$. Conversely, let graph G_{TS} have a subgraph G^+ such that every node $w \in W$ is $k - trackable$. Then, each of the q mobile nodes in G^+ is connected with a clique of k vertices from the set V. Each of these cliques is disjoint (no shared vertices) and only contains edges from the set E. Hence there is a partition of the graph $G = (V, E)$ into q disjoint cliques of size k. This concludes the proof. □

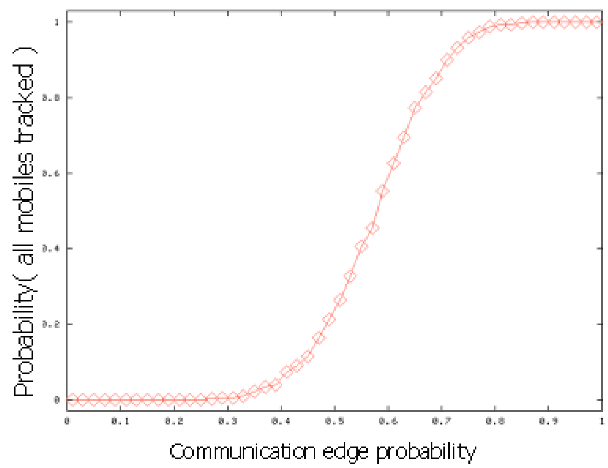

Figure 4: Phase Transition in the probability of obtaining a solution to SensorCSP with respect to the communication edge probability model

In this problem the computational complexity is closely related to the communication level between the sensor nodes. Our preliminary results indicate that, as the number of communication edges is increased, this problem also shows the phase transition phenomenon that has been identified in many difficult computational problems [1, 7].

Figure 4 shows the result obtained with experiments with a random configuration of 9 sensors and 3 mobile nodes such that there is a communication channel between two sensors with probability p. In these experiments all mobile nodes were assumed to be within range of all sensors, and the number of required sensor trackers k is equal to 3. The plot shows the probability that three communicating sensors can be assigned to track each mobile node. On the far left hand side of the phase transition, when p is near 0, there are few communication edges and it is hard to find such an assignment for every mobile node. On the other hand, when p is near 1, the system has almost a global communication property (nearly all pairs of sensors can directly communicate with each other), and it is almost certain that there exists such an assignment. As with other CSPs, it is expected that the harder instances of SensorCSP are likely to occur around the region of the phase transition.

We note that the results in figure 4 are based on a random graph model, and in fact the phase transition that we observe in this figure is the one expected for monotone first order properties in random graphs [9].

3 Distributed Constraint Satisfaction Model

In the wireless radar tracking system, it is assumed that computational resources are available at each sensor node for the instantiation of agents. It is desirable for the agents to solve the overall problem in a distributed manner by making use of local information. For modeling this, we consider formulating SensorCSP as a distributed constraint satisfaction problem formulation (DCSP) [13].

In a DCSP, variables and constraints are distributed among multiple agents. As explained in [13], it consists of a set of agents $A = \{a_1, a_2, \ldots a_n\}$, each of which has its own variables. There are constraints on the assignments to variables of each agent (intra-agent constraints), as well as constraints on the assignments for variables of different agents (inter-agent constraints). A solution to this problem is an assignment to the variables of all the agents such that all these constraints are satisfied.

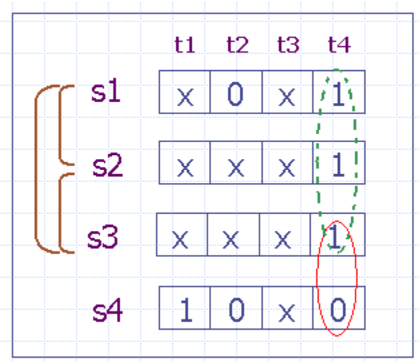

Figure 5: Sensor-centered Agents: Variables and Constraints

In this problem domain, one can think of having a distinct agent for each sensor, or having a distinct tracker agent for each mobile node[1]. Each of these views results in a different DCSP model: a sensor-centered model, and a mobile node-centered model. We describe these two models below in more detail.

3.1 Sensor-centered DCSP

In this model, each sensor agent has a binary variable corresponding to each mobile node that is within range of it. A sensor agent assigns a value 1 to a variable t if it is going to track the mobile node corresponding to this variable.

Since each sensor can track at most one in-range mobile node, at most one of its variables can be set to 1. This is the intra-agent constraint in this model. In SensorCSP, we desire that k communicating sensors should track each mobile node - this forms the basis of the inter-agent constraints.

Figure 5 shows an example variable assignment for the sensor-centered DCSP with $k = 3$. Each row represents the variables of a particular sensor, in such a way that the variables corresponding to the same

[1] We assume that in either case the agents reside physically in sensors only, where the computational resources are.

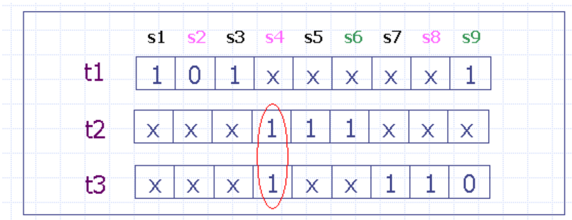

Figure 6: Mobile node-centered Agents: Variables and Constraints

mobile node are placed in the same column. In each row, the elements corresponding to out-of-range mobile nodes are crossed out since these variables are not available to the sensor agent. For example, row one corresponds to the sensor agent s1 which can assign values to variables corresponding to mobile nodes t2 and t4. In the current assignment it has chosen to assign a 1 to the variable for t4. In this example, sensors s1, s2, and s3 are able to communicate with each other but not with s4. Hence the first three sensor agents can all decide to track mobile node t4 since the inter-agent constraint depicted by the dashed-line oval is satisfied. Another inter-agent constraint is that sensor agent s4 cannot choose to track mobile node t4 because there is no communication between s3 and s4, and this is depicted by the solid-line oval.

3.2 Mobile node-centered DCSP

In this model, there is a tracker agent associated with each mobile node. Each tracker agent has a binary variable corresponding to each sensor that has the mobile node within its range. A tracker agent t assigns a value 1 to a sensor variable s if that sensor is going to track the mobile node corresponding to t.

In SensorCSP, it is required that each mobile node be tracked by k communicating sensors that it is within range of. This forms the basis of the intra-agent constraints - each tracker agent must select exactly k sensor variables to be 1, and these k sensors must be able to communicate with each other in a pairwise manner. The inter-agent constraint in this model is that no two tracker agents may select the same sensor. This is because of the assumption that each sensor can only track one mobile node at the same time.

Figure 6 shows an example variable assignment for the mobile node-centered DCSP. Each row represents the variables of a particular mobile tracker agent, in such a way that the variables corresponding to the same sensor node are placed in the same column. In each row, the elements corresponding to sensor nodes that cannot track this mobile node are crossed out since these variables are not available to the mobile tracker agent. For example, row one corresponds to the mobile tracker agent t1 which can assign values to variables corresponding to sensors s1, s2, s3, and s9 (since $k = 3$). In the current assignment agent t1 has chosen to assign a 1 to the variables for s1, s3, and s9. In this example, agents t2 and t3 have a constraint violation because they have both assigned a 1 to their respective variables corresponding to sensor s4. This inter-agent constraint violation is depicted by the solid-line oval in the figure.

We assume that each mobile tracker agent is located in one of the sensors that the mobile node is within range of. We also assume without loss of generality that this agent is able to communicate directly with the other sensors that can potentially track the same mobile node [2]. Tracker agents for two mobiles can communicate directly with each other through common sensor nodes that are within range of both mobiles.

3.3 Comparison of the two DCSP models

Table 1 compares the two models. We observe that in both models we have the constraints "Only one mobile per sensor" and "k communicating sensors per

[2]Even if this is not the case, the tracker agent will only consider assigning communicating sensors to track the mobile node.

DCSP Model	**Sensor-centered**	**Mobile node-centered**
Agents	Sensors (s)	Mobiles (m)
Variables	Mobiles (m)	Sensors (s)
Intra-agent Constraints	Only one mobile per sensor	k communicating sensors per mobile
Inter-agent Constraints	k communicating sensors per mobile	Only one mobile per sensor

Table 1: Dual models for distributed SensorCSP

mobile". The difference is that in the sensor-centered model the first one is a intra-agent constraint and the other is a inter-agent constraint, but in the mobile node-centered model the situation is exactly the reverse. This difference can be significant from the point of view of solution, because DCSP algorithms must make use of the agents' communication resources in order to check that the inter-agent constraints are satisfied. Hence the solution of a DCSP formulation which involves more complex inter-agent constraints is likely to incur a greater communication cost.

It is expected that, in realistic settings, the number of sensors is much greater than the number of mobile nodes to be tracked. Under these circumstances the number of sensors that can possibly track a given mobile node is likely to be greater than the number of mobile nodes within range of each sensor. Thus the number of variables in the two different DCSP models can be quite different. Sensor-centered agents would have typically fewer variables than mobile node-centered agents, but there will be more agents in total in the former case. When each agent has fewer variables, its computational requirements are reduced since there are fewer intra-agent constraints to be satisfied. In fact, in the case of the sensor-centered model, the intra-agent constraints are very easy to satisfy since it only requires each sensor agent to assign a 1 to exactly one of its variables.

Thus we see that there is a tradeoff between the communication and computational complexity in the two formulations. The sensor-centered model may require less computation at each node, at the expense of greater communication costs, as compared to the mobile node-centered model.

4 Distributed Solution of SensorCSP

Complete search algorithms to solve constraint satisfaction problems are based on backtracking [3]. Recently such algorithms have been modified for use in solving distributed CSPs as well. Two such approaches are the asynchronous backtracking algorithm (ABT) [13] and the distributed backtracking algorithm (DIBT) [4]. These are both known to be sound and complete algorithms.

In these algorithms, a priority order is first induced among the distributed agents. When solving the problem, each agent tries to solve its own CSP first and inform its neighboring agents of the current assignment of its variables . Lower priority agents are expected either to modify their current assignment to be consistent with the assignments and notify their neighbors in turn, or if this is not possible, to request higher priority agents to perform a backtrack step.

We make a distinction in SensorCSP between explicit and implicit inter-agent constraints. *Explicit* inter-agent constraints are the ones that are originally defined in the problem. For the mobile-node centered approach, the explicit inter-agent constraint between two tracker agents is that they may not both assign a 1 to variables corresponding to the same sensor. We assume in our test-bed that the tracker agents for these mobile nodes will communicate through the set of common sensors, hence all agents with explicit constraints can communicate directly with each other. *Implicit* constraints are those that are implied by a set of explicit constraints. What is different about this test-bed is that two agents with implicit constraints may not necessarily be able to communicate with each other directly.

Consider the example shown in figure 6, which shows the mobile node-centered representation for distributed SensorCSP. In this example, therefore, the tracker agents for mobiles t2 and t3 have an explicit constraint that they cannot simultaneously assign sensor s4 to track them, and the tracker agents for t1 and t3 have an explicit constraint that they cannot simultaneously assign sensor s9 to track them. There is no explicit constraint between the tracker agents for t1 and t2, since there is no common sensor between them.

However, there is an implicit constraint between agents t1 and t2. This can be seen as follows: in the figure, currently t2 has a conflicting assignment with t3, but is unable to resolve this conflict because the only possible modification creates a conflict between t3 and t1. Thus, indirectly, there is a conflict between the current assignments of t2 and t1. Thus there is an implicit constraint between t2 and t1 that is implied by the two explicit constraints in this situation. Observe that agents t1 and t2 in this situation cannot communicate directly with each other since they do not share any common sensors.

In ABT, such implicit constraints may be discovered between two agents during the execution of the algorithm. In this case, the ABT algorithm allows the agents to create a constraint link to each other. It is stated in [13] that "since a link in the constraint network represents a logical relationship between agents, adding a link does not mean adding a new physical communication path between agents." Similarly, in DIBT, it is assumed that all agents which have constraints between them, explicit or implicit, can communicate directly with each other.

In our domain, communication between agents may not be straightforward. It is true that because of the way the implicit constraints are discovered between two agents, there is guaranteed to be a chain of intermediate agents which will provide a communication path between them. But this will require the routing of information through multiple sensors, increasing the communication cost. Thus in studying communication complexity in this domain, we need to make a careful distinction between explicit and implicit constraints.

This issue can have a big impact on the scalability of the distributed system. One can envision a situation where two tracker agents that are relatively far apart in a large network have an implicit constraint between them. Communication between these agents, which may be necessary to find the globally satisfying solution, could involve an unacceptably large utilization of the network's communication resources. One of our goals in comparing the two DCSP models is to see in both cases what role the implicit constraints play in their communication complexity.

A related question is what happens if the constraint graph formed by the explicit constraints has a "small world" topology [12]. In other words what happens when the nodes in the graph of explicit constraints are highly clustered and we can find an implicit constraint between any two agents that is implied by a relatively short chain of linked explicit constraints? Some recent work [10] has suggested that under small world topologies the solution of a constraint satisfaction problem becomes hard because local changes in the values of some variables impact the assignments of a large number of variables.

5 Conclusions and Future Work

In this paper, we have presented SensorCSP, an interesting problem domain for studying distributed constraint satisfaction. We showed that this problem is NP-complete, and that the communication between sensors in this system is a key factor in its combinatorial complexity. We showed that this problem has two DCSP representations that can be seen as duals. We believe that these two representations effect different tradeoffs between the communication and computational complexity. We also discussed the difference between explicit and implicit constraints in SensorCSP and how this might impact the scalability of the distributed system.

We would like to emphasize that this is ongoing work. We are currently in the process of applying versions of known sound and complete DCSP algorithms such as the asynchronous backtracking algorithm and the distributed backtracking algorithm to this problem to get a quantitative picture of the complexity issues from a distributed agent point of view. There has been some recent work showing phase transitions in DCSP [5], and it would be interesting to see if similar results can be obtained with SensorCSP when it is solved in a distributed fashion. As we discussed earlier, the dual DCSP representations of SensorCSP each have different numbers of inter-agent and intra-agent constraints. We would also like to see how this affects the communication and computational complexity of the two approaches.

There are some interesting extensions of this work that we would like to explore in the future. One question that remains to be answered is how the problem complexity might be affected by the model used to generate the topology of the SensorCSP graph G^*.

The wireless sensor tracking network that SensorCSP is based on, is of course, a real-time system. We would like to extend our work to considering the dynamic version of this problem. When the mobile nodes are moving, the graph G* and the corresponding constraints change with time. Even if there are several satisfying assignments for the static case, one of these might be more robust with respect to small changes in the underlying graph. What kind of online distributed constraint satisfaction algorithms are required under these circumstances?

SensorCSP is clearly a promising benchmark problem for distributed constraint satisfaction. We have discussed a number of interesting issues that arise in this domain. We are currently in the process of obtaining some concrete experimental results concerning these issues.

References

[1] P. Cheeseman, B. Kanefsky, and W. M. Taylor. Where the really hard problems are. *IJCAI-91*, 1:331–7, 1991.

[2] S. E. Conry, K. Kuwabara, V. R. Lesser, and R. A. Meyer. Multistage negotiation for distributed constraint satisfaction. *IEEE Transactions on Systems, Man, and Cybernetics (Special Section on DAI)*, 21(6):1462–1477, 1991.

[3] R. Dechter and D. Frost. Backtracking algorithms for constraint satisfaction problems. Technical report, http://www.ics.uci.edu/~csp/r56-backtracking.pdf, Information and Computer Science Department, UC Irvine, 1999.

[4] Y. Hamadi, C. Bessière, and J. Quinqueton. Backtracking in distributed constraint networks. In Henri Prade, editor, *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98)*, pages 219–223, Chichester, August 23–28 1998. John Wiley & Sons.

[5] K. Sycara K. Hirayama, M. Yokoo. The phase transition in distributed constraint satisfaction problems: First results. In *Proceedings of the International Workshop on Distributed Constraint Satisfaction*, 2000.

[6] D.G. Kirkpatrick and P. Hell. On the complexity of general graph factor problems. *SIAM Journal of Computing*, 12(3):601–608, 1983.

[7] Remi Monasson, Riccardo Zecchina, Scott Kirkpatrick, Bart Selman, and Lidor Troyanski. Determining computational complexity from characteristic phase transitions. *Nature*, 400(6740):133–137, 1999.

[8] Sanders and Air Force Research Lab. ANTs challenge problem. *http://www.sanders.com/ants/overview-05-09.pdf*, 2000.

[9] J. Spencer. *Ten Lectures on the Probabilistic Method.* SIAM, Philadelphia, PA., 1987.

[10] Toby Walsh. Search in a small world. In Dean Thomas, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99-Vol2)*, pages 1172–1177, S.F., July 31–August 6 1999. Morgan Kaufmann Publishers.

[11] William E. Walsh and Michael P. Wellman. MarketSAT: An extremely decentralized (but really slow) algorithm for propositional satisfiability. In *Proceedings of the 17th Conference on Artificial Intelligence (AAAI-00)*, pages 303–309. AAAI Press, July 30 - August 3, 2000.

[12] D. Watts and S. Strogatz. Collective dynamics of small-world networks, Nature 393 (1998) 440–442, 1998.

[13] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):673–85, 1998.