Towards Lightweight Self-Configuration in Wireless Sensor Networks

Benjamin Satzger, Faruk Bagci, Florian Kluge, Theo Ungerer
Department of Computer Science
University of Augsburg, Germany
{satzger, bagci, kluge, ungerer}@informatik.uni-augsburg.de

ABSTRACT

This paper proposes a new algorithm called SDS for solving distributed constraint satisfaction problems in wireless sensor networks. It introduces minimal overhead by using piggybacking for information dissemination and is naturally insusceptible against message loss. Partitioning into coordinating cliques has been used to demonstrate the feasibility of our approach.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Algorithms

Keywords

self-configuration, wireless, constraint satisfaction problem (CSP), distributed CSP (DCSP)

1. INTRODUCTION

Constraint satisfaction problems (CSPs) [3] are subject to research in the areas of artificial intelligence and operations research. They are defined as a set of objects which must satisfy a number of constraints. CSPs conform to a simple standardized representation and allow to apply generic, domain-independent solution algorithms. Distributed constraint satisfaction problems (DCSPs, also DisCSPs) represent a formalism for describing a problem that involves multiple participants called agents. DCSPs have first been investigated by Yokoo et al. [4]. They can be used to model many real-world problems with distributed nature, such as self-configuration in networks. In this paper we propose a DCSP algorithm which is especially suited for wireless networks. It does not send any message but only appends small bits of information to messages sent anyway. Compared to traditional DCSP solvers which rely on active message transmission the proposed algorithm SDS introduces minimal overhead in terms of energy consumption and does not cause message congestion and collisions.

SAC'10 March 22-26, 2010, Sierre, Switzerland.

2. CONSTRAINT SATISFACTION PROBLEMS

A CSP is defined by a set of variables, $V = \{X_1, X_2, \ldots, X_n\}$, and a set of constraints, $C = \{C_1, C_2, \ldots, C_m\}$. Each variable X_i has a non-empty domain D_i of possible values. Each constraint C_i involves some subset of the variables and specifies the allowable combinations of values for that subset. A state of the problem is defined by an assignment of values to some or all of the variables, $\{X_i = v_i, X_j = v_j, \ldots\}$. An assignment that does not violate any constraint is called *consistent* or *legal*. A *complete* assignment is one in which every variable is mentioned, and a solution to a CSP is a complete and consistent assignment.

3. DISTRIBUTED CONSTRAINT SATISFACTION PROBLEMS

A DCSP is a generalization of a CSP where variables are distributed among agents $A = \{A_1, \ldots, A_l\}$, i.e. each agent *owns* some variables. The most trivial approach to solve a DCSP is to select a leader agent responsible to collect all information about variables, their domains and all constraints. In this case the leader is able to use a standard centralized CSP algorithm to solve the DCSP. This approach, however, suffers amongst others from privacy issues. The probably most well known distributed, asynchronous algorithm for solving DCSPs has been introduced by Yokoo et al. [4].

4. SYSTEM-DRIVEN SEARCH

We assume that agents are autonomic computing entities running on a stationary wireless node, i.e. a device with a wireless interface to send and receive messages "over the air". A simple *unit disk graph*, an intersection graph of equal-radius circles, is used to model the topology of the wireless network. Due to the nature of wireless communication we suppose that a message sent by an agent is received by all its direct neighbors. Wireless networks like WSNs typically have strict energy constraints and have to deal with limited battery capacity. Communication amongst nodes consumes the largest part of energy [1]. The algorithm we are proposing does not send any message but only appends small bits of information to messages sent anyway. Thereby, communication introduced by solving DCSPs is avoided which results

in less energy consumption and improved network lifetime. As the algorithm does not actively transmit data but depends on the system activities we call our DCSP algorithm for wireless networks *System-Driven Search* (SDS).

Every agent appends information about its current status to every message sent by the node it is running on. When an agent receives a message it analyzes the sender's status and adapts its variable assignments in order to satisfy its constraints. Every agent needs to know its variables V and their corresponding domains, its constraints C and the current values assigned to its variables VA. A set *agentview* serves as a store for information received from other agents which is initially empty. There are three events upon which the algorithm reacts. On start-up, the agents assign random values to their variables which are consistent with their intra-agent constraints. When the node is sending a message the agents append the triple (V, C, VA) to the message, i.e. its variables, constraints and assignments. As variables and constraints do not change over time it would be sufficient to transmit this data once and only append updated assignments in the course of the algorithm. Upon receipt of a message the agent updates its *agentview* with the appended data. If it receives a message from an agent for the first time a new entry (V, C, VA) is inserted into *agentview*. Next time only the variable assignments are updated. After that, it is checked whether the current assignments VA are consistent with all values in *agentview* and its own constraints. If this is the case nothing needs to be done. If the current assignment is inconsistent, it is searched for a random consistent assignment. If the search is successful the assignment is being adjusted. If *agentview* does not allow a consistent variable assignment, *agentview* is cleared except for the latest entry and it is tried to find a solution which at least conforms to the latest entry. In the worst case the latest entry of *agentview* is also deleted and a random assignment is applied which is only consistent with the intra-agent constraints.

5. EVALUATION

For evaluation purposes we have applied SDS to a specific configuration problem typically arising in wireless sensor networks: The partition into subgroups of completely interconnected nodes. This problem, which is NP-hard in its general form, arises for instance if collections of sensor nodes are responsible for the joint execution of certain tasks, like tracking an object [2]. For our measurements we have restricted the problem to the formation of cliques of size three. We consider arbitrarily positioned nodes within a plane of size 1000x1000. Hereby we vary the number of nodes n (9,15,24) and their transmission radius R (300,400,500,600). Each single scenario is repeated 1000 times using the same setting to obtain representative averaged results. The SDS algorithm is based on the fact that communication among nodes is taking place. The simulation is conducted in rounds and in each round the probability of a node to send an application message is 10%. Note that SDS does not rely on a round based system nor takes advantage of that fact. SDS uses application messages for information dissemination. Every experiment starts with the random placement of nodes. Then, a centralized algorithm checks whether the generated problem instance is solvable or not. SDS has been applied only to solvable problem instances. The simulation environment counts the number of sent application messages

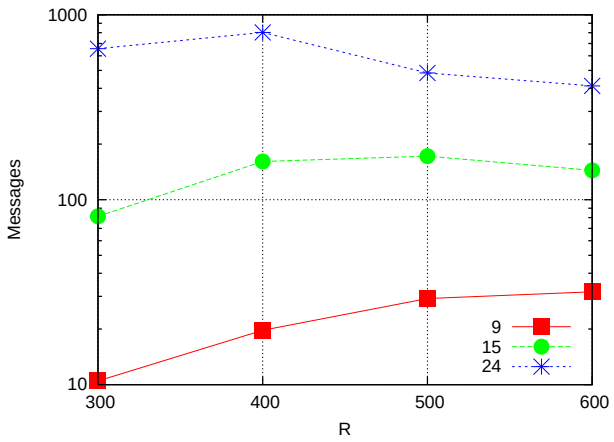

Figure 1: Messages for partitioning into cliques

until a valid solution has been found using SDS. Every solvable problem instance has been successfully solved by SDS. Figure 1 shows the number of randomly sent application messages until SDS has found a solution to the DCSP.

6. CONCLUSION

In this paper, we propose a new algorithm called SDS for solving DCSPs in wireless environments. SDS uses application messages for information dissemination instead of actively transmitting data. The resulting savings of energy on the one hand and avoidance of congestion and message collisions on the other hand are a unique feature of SDS compared to other algorithms known to the authors. Especially for solving uncritical DCSPs, SDS could be the first choice. For problem instances where it is not guaranteed that a solution exists, e.g. due to disadvantageous spatial distribution of nodes, traditional solvers would consume much resources just to find out that there is no solution. Due to its nature SDS is very robust against message losses which sets SDS apart from many traditional DCSP solvers which fail if one single message gets lost. However, SDS cannot guarantee that a solution for solvable problems is found, as it relies on application messages which cannot be influenced actively. In the worst case if no single application message is sent nothing can be achieved.

7. REFERENCES

[1] S. G. Akojwar and R. M. Patrikar. Improving Life Time of Wireless Sensor Networks Using Neural Network Based Classification Techniques With Cooperative Routing. *International Journal of Communications*, 2(1), 2008.

[2] B. Krishnamachari, R. Bejar, and S. Wicker. Distributed problem solving and the boundaries of self-configuration in multi-hop wireless networks. *HICSS*, 9:297b, 2002.

[3] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2002.

[4] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara. Distributed Constraint Satisfaction for Formalizing Distributed Problem Solving. In *ICDCS*, pages 614–621, 1992.