Distributed Constraint Satisfaction and the Bounds on Resource Allocation in Wireless Networks

Bhaskar Krishnamachari, Rámon Béjar*, and Stephen Wicker
School of Electrical and Computer Engineering
Cornell University, Ithaca, New York 14853
{bhaskar@ece, bejar@cs, wicker@ece}.cornell.edu

Abstract

In this paper we consider medium access scheduling in ad hoc networks as a distributed constraint satisfaction problem (DCSP), and present experimental results on the solvability and complexity of this problem. We show that there are "phase transitions" in solvability and complexity with respect to the transmission power of the wireless nodes. The phase transition curves indicate that there is a critical maximum power level for certain arrangements of nodes and a given availability of spectrum in an ad hoc network beyond which the problem of channel allocation becomes intractable.

1 Introduction

In this paper we consider a specific instance of an NP-hard constraint-satisfaction problem that arises naturally in the context of ad hoc wireless networks. Constraint satisfaction is a useful formalism for modelling a large class of problems with applications in engineering design, planning, scheduling, resource allocation, fault diagnosis [6]. In a constraint satisfaction problem (CSP), there are a number of variables, each of which has an associated domain of values. A number of constraints are specified on subsets of these variables restricting the set of values they can take on jointly. The objective of the problem is to find out if each of these variables can be assigned a value from its domain in such a way that all the constraints are satisfied.

*This author is with the Intelligent Information Systems Institute, Department of Computer Science, Cornell University.

The original NP-complete problem, satisfiability (SAT) [8], is a special kind of CSP. We briefly describe SAT for the purpose of illustration. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of Boolean *variables*. Each variable x_i and its negation $\overline{x_i}$ constitute *literals*. A *clause* is a disjunction (OR) of one or more literals (e.g. $(x_1 \vee \overline{x_2})$) and is said to be *satisfiable* if there exists some *truth assignment* of 0/1 values to all variables such that at least one of its literals evaluates to true under that assignment. Two special cases are the unit clause, represented (l), that contains only one literal, and the *empty clause*, represented $(\Box)$, which contains no literals and is by definition unsatisfiable. A conjunctive normal form (CNF) formula over X consists of the conjunction (AND) of a number of clauses, and is said to be satisfiable if there exists some truth assignment to the variables in X such that all the clauses are satisfied.

An instance of SAT consists of a CNF formula Γ with the goal to determine if there exists a satisfying truth assignment for Γ. For example, the formula $\Gamma = (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2)$ is satisfied by setting both x_1 and x_2 to 1; the formula $(x_1) \wedge (\overline{x_1})$ is unsatisfiable since only one of the clauses can be satisfied by setting x_1 to either 0 or 1. SAT is a constraint satisfaction problem as the clauses in the formula represent constraints on the Boolean variables.

For many CSPs, including SAT, it is known that as the ratio of constraints to variables is increased, the fraction of (randomly generated) instances that are solvable undergoes a one to zero "phase transition" [3, 11, 14]. Further, the computational cost of determining whether or not an instance is satisfiable shows an easy-hard-easy pattern, with the complexity peaking in the phase transition region. It is easy to solve CSPs when they are under-constrained, and easy to show that they have no solution when they are over-constrained. The hardest instances lie in the critically-constrained phase transition region. In the context of engineering design, it is our goal to incorporate into adaptive systems those network configuration problems that fall into the "good," under-constrained region of the complexity spectrum.

In the remainder of this paper we consider ad hoc networks that are characterized by randomly placed nodes with a fixed transmitter power level. Given the assumption of limited spectral resources, we show that the problem of medium access scheduling, when interpreted as a distributed constraint satisfaction problem, exhibits the same phase transitions as the SAT problem. We conclude that there is a maximum power level for the nodes beyond which the problem of medium access scheduling becomes over-constrained and intractable.

2 DCSPs

A Distributed Constraint Satisfaction Problem (DCSP) [18], is a generalization of a CSP to the framework of distributed problem solving. In a DCSP, there are a set of n Agents $A = \{1, 2, \cdots, n\}$. Each agent has its own variables with corresponding domains for each of them. There are *intra-agent* constraints between the variables of each individual agent, and *inter-agent* constraints between the variables of different agents. A solution to the DCSP is an instantiation of values to all the variables of each agent such that every intra and inter-agent constraint is satisfied.

For solving a DCSP, in order to satisfy the inter-agent constraints, agents need to use some communication mechanism for exchanging the value of their variables to other agents. Thus in addition to the computational effort expended at each node, we also need to consider the communication complexity involved in solving a DCSP. One measure of the communication complexity for a DCSP is the number of messages exchanged by the agents in order to solve the problem or to detect that no solution exists.

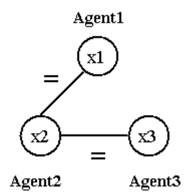

Figure 1: Satisfiable Boolean DCSP

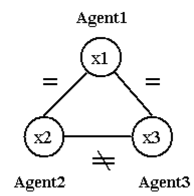

Figure 2: Unsatisfiable Boolean DCSP

Figure 1 gives an example of a satisfiable DCSP, i.e. one that has at least one solution. This DCSP consists of three agents, it has one variable for each agent and there are inter-agent constraints, but no intra-agent constraints. The domain of all the variables is $\{0, 1\}$. The interagent constraints are represented in the figure as edges with a binary relation symbol. The relation symbol specifies the relation that must hold between the variables of the two connected agents. A possible solution for this DCSP is for all agents to set the same value (0 or 1) to their variables. Figure 2 gives an example of an unsatisfiable DCSP. There is no possible solution for this DCSP, because the fact that $x_1 = x_2$ and $x_1 = x_3$ must be true *implies* that $x_2 = x_3$ should also be true, which would violate the inter-agent constraint between agents 2 and 3.

An algorithm is said to be *complete* if, when applied to a given problem, it either provides a solution to the problem or determines that the problem has no solution. Two complete algorithms have been developed for solving any distributed problem that can be formalized as a DCSP: the distributed backtracking algorithm (DIBT) [9] and the asynchronous backtracking algorithm (ABT) [18]. These two algorithms work by using a generalization of systematic complete search in the distributed setting. Because we are working in an asynchronous environment (there is no central control) all the agents can decide by themselves when to change the values assigned to their variables.

At the beginning, both algorithms proceed as follows. All the agents choose a value for all their variables such that their intra-agent constraints are satisfied [1] Before the search can proceed, we need also to assign an unique identifier number to every agent. This identifier is used to establish a priority order between agents. One agent has a greater priority than other if its identifier is smaller. Given a inter-agent constraint between two agents, the highest priority agent has priority to change the value of a variable that appears in the constraint (and that belongs to him). It must inform the other agent about any change to the variable by sending an *information* message. When the other agent receives the information message, it must try to find an assignment to its own variables such that all the inter-agent constraints that it has with higher priority agents, and its own intra-agent constraints, are satisfied. If it changes the value of some of its variables, it will send information messages to all its own, lower priority, agents such that they have some inter-agent constraint with him. However, if there is no possible assignment consistent with the inter-agent constraints of higher priority agents, it will send a *backtracking* message to the lowest priority agent among all its higher priority agents that have an inter-agent constraint that is not satisfied. This message tells the higher priority agent that it must try to find a different value for the variable that is causing a conflict with the lower priority agent, because the lower priority agent cannot do anything for fixing the conflict.

[1] They can achieve this using any existing centralized CSP algorithm.

For the details of the specific algorithms, the reader is referred to the references [9] [18].

3 DCS in Wireless Networks

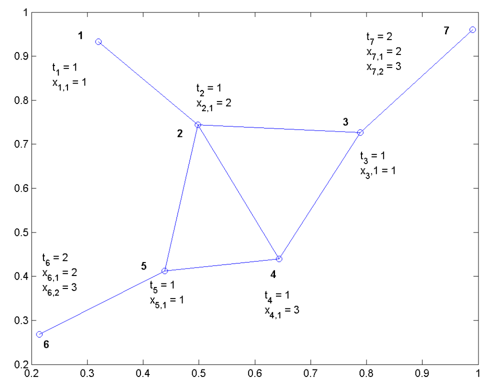

Figure 3: Solvable channel scheduling with small transmission radius (n =7, C = 3, R = 0.40)

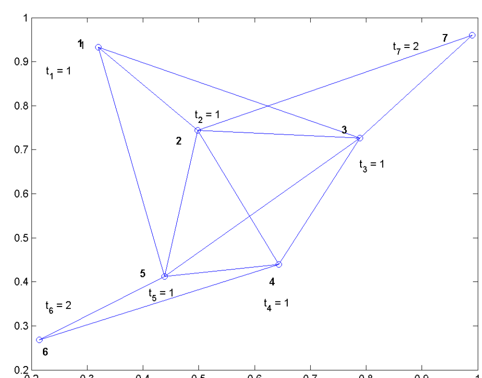

Figure 4: Unsolvable channel scheduling with large transmission radius (n =7, C = 3, R = 0.55)

The multiple access problem arises in wireless ad-hoc networks because transmissions by one node can interfere with transmissions by nearby nodes. Multiple

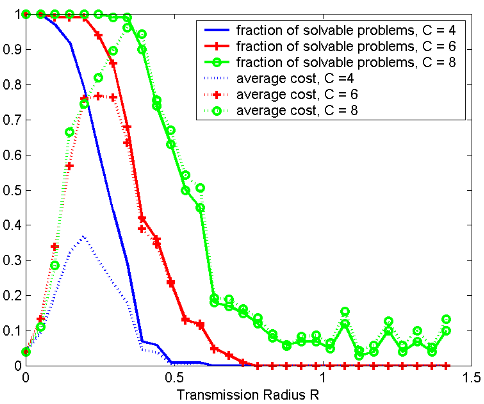

Figure 5: Phase transitions in the fraction of solvable problems and the average complexity for the channel scheduling problem using a complete backtracking algorithm

access techniques can be categorized as random access, scheduled access, or hybrids. Most randomized access algorithms proposed for ad-hoc networks are variations of ALOHA [1] and Busy Tone Multiple Access (BTMA) [17], both of which are useful for bursty traffic conditions. In these techniques, nodes share the same broadcast channel and transmit whenever they need to. The scheduled access techniques that have been proposed for ad-hoc networks [12, 15, 2] are better suited for non-bursty traffic conditions. In scheduled access techniques the available bandwidth is typically divided into multiple time, frequency, or code division multiple access channels. Each node schedules its transmission on different channels in such a way as to avoid conflicts with neighboring nodes and achieve efficient spacial reuse. The problem of scheduled access in ad-hoc networks in general is NP-hard [15].

Consider an ad-hoc wireless network consisting of n nodes each of which transmits with the same power. We will assume that the transmission range of each node can be modelled as a circle of some radius R centered at the node. Let each node i in the network have a specified traffic need for t_i contiguous time channels. A total number of C channels are available. The goal is to find an assignment of t_i time channels to each node i, such that no two neighboring nodes j and k share the same channel. This can be easily modelled as a DCSP. Imagine each node as an agent, with t_i multi-valued variables $\{x_{i,1}, \ldots x_{i,t_i}\}$for each agent i, corresponding to the allocated channels. These variables can take on values from 1 to C. The intra-agent constraint here is that each of these variables within an agent must take on distinct values. The inter-agent constraints is that if there are two neighboring (interfering) nodes i and j, their variables must not take on the same values.

Formulated as a DCSP, this problem can be solved using one of the distributed backtracking algorithms mentioned in the previous section. Although the communication and computational costs involved can be exponential in the number of nodes in the worst case, as we have discussed before, the average complexity can be within tolerable limits provided the system as a whole is under-constrained.

Figure 3 shows a solvable instance of this problem on a small, sparse network. An assignment to the variables of each node agent that satisfies all constraints is indicated in the figure. Figure 4, on the other hand, is an unsolvable instance of this problem on a dense network. Since there are only three channels available, and the nodes 2,3,4, and 5 form a clique of size 4, it is not possible for them to assign values to their respective variables without violating inter-agent constraints.

For a given average traffic per node, there are two parameters that affect the problem complexity and solvability: the transmission radius R, and the total number of channels available C. In order to study the effects of these parameters, we conducted the following experiment. 7 nodes are placed at random in a square region with unit sides. The transmission radius of all nodes is the same and is varied from 1 to $\sqrt{2}$. A particular combination of node positions and transmission radius corresponds to a unique network graph. A traffic of 1 or 2 is generated at each node with equal probability. The bandwidth C is tested

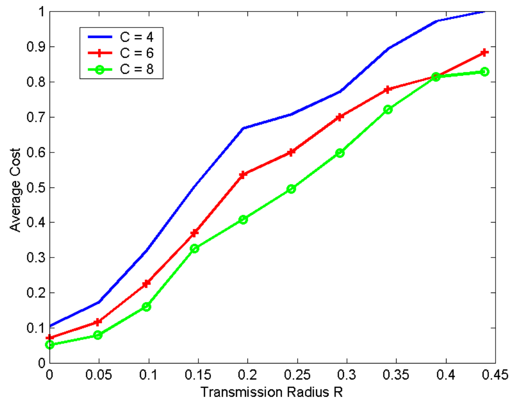

Figure 6: average computational complexity of solving the channel scheduling problem for satisfiable instances using a randomized local search algorithm

for values 4,6, and 8. 100 instances are generated for each value of transmission radius and bandwidth. A complete CSP-solver was used to obtain statistics on satisfiability and computational complexity of these instances. Figure 5 contains the results of these experiments, and shows that this problem undergoes a phase transition with respect to the transmission radius[2]. There is a critical value of the transmission radius below which a satisfying solution exists with high probability, and above which it exists with negligible probability. Figure 5 also shows the easy-hard-easy phase transition in average complexity when this constraint satisfaction problem is solved using a backtracking algorithm. It is interesting to see that when we increase the total number of channels, the bandwidth available to the system - the phase transition threshold moves to the right. This is intuitive – adding bandwidth resources to this system makes it easier to provide a non-conflicting schedule to the nodes.

[2] Note that in this problem the transmission radius increases with the ratio of constraints to variables. This is because for a given level of traffic, number of nodes, and number of channels, the number of variables is fixed, and the number of constraints increases with the density of the network graph which in turn depends directly upon the transmission radius.

Another experiment makes it clear that we can also expect some gains in complexity to result from the increase in bandwidth. If we restrict the study of complexity to only those instances in which there is a solution, i.e. satisfiable instances, then we can use a randomized local search algorithm to solve the problem. Figure 6 shows the normalized average cost for finding the satisfying solution for the three values of the bandwidth C ranging from 4 to 8. For each value of the transmission radius , the cost (which is a measure of the time taken to obtain the solution) is averaged over 300 runs of the local search algorithm, each starting from a random point in the search space. Figure 6 reveals that an increase in bandwidth not only increases the probability that a solution, but also decreases the complexity of obtaining the solution when it exists. In short, if we use our knowledge of the fundamental limits of the resource allocation problem to narrow the focus of the search, we can take advantage of suboptimal random search algorithms to obtain a rapid and effective solution.

4 Discussion and Conclusions

In this paper we have shown that one of the NP-hard problems that underlie the design of self-configuring ad hoc networks can be characterized in terms of a critical transition in complexity. In the case studied in the paper, it was seen that the number of channels needed by the network to support a medium access protocol was critically dependent on the transmitter power level of the individual nodes. In general, truly scalable networks require the availability of a critical amount of bandwidth and energy resources.

References

[1] N. Abramson. The aloha system – another alternative for computer communications. In *Proceedings of Fall Joint Computer Conference*, 1970.

[2] L. Bao and J. J. Garcia-Luna-Aceves. Collision-free topology-dependent channel access scheduling. In

MILCOM 2000. 21st Century Military Communications Conference Proceedings, pages 507–511, 2000.

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

[4] Scott H. Clearwater, Bernardo A. Huberman, and Tad Hogg. Cooperative solution of constraint satisfaction problems. *Science*, 254:1181–1183, November 22 1991.

[5] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5:394–397, 1962.

[6] 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.

[7] Deborah Estrin, Ramesh Govindan, John S. Heidemann, and Satish Kumar. Next century challenges: Scalable coordination in sensor networks. In *Mobile Computing and Networking*, pages 263–270, 1999.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, San Francisco, 1979.

[9] 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.

[10] 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.

[11] S. Kirkpatrick and B. Selman. Critical behavior in the satisfiability of random Boolean expressions. *Science*, 264(5163):1297–1301, 27 May 1994.

[12] V.O.K. Li L.C. Pond. A distributed time-slot assignment protocol for mobile multi-hop broadcast packet radio networks. In *MILCOM*, pages 70–74, 1989.

[13] N.A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, Inc., San Francisco, California., 1996.

[14] D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of sat problems, 1992.

[15] K.K. Parhi R. Ramaswami. Distributed scheduling of broadcasts in a radio network. In *INFOCOM*, pages 497–504, 1989.

[16] Bart Selman, Henry A. Kautz, and Bram Cohen. Noise strategies for improving local search. In *Proc. of the National Conference for Artificial Intelligence (AAAI'94)*, pages 337–343, 1994.

[17] F.A. Tobagi and L. Kleinrock. Packet switching in radio channels, part ii: The hidden terminal in carrier sense multiple access and the busy-tone solution. *IEEE Transactions on Communication*, 23:1417–1453, 1975.

[18] 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.