A Distributed Constraint Optimization Approach for Coordination under Uncertainty

James Atlas*
University of Delaware

Keith Decker†
University of Delaware

Abstract

Distributed Constraint Optimization (DCOP) provides a rich framework for modeling multi-agent coordination problems. Existing problem domains for DCOP focus on small (<100 variables), deterministic domains. We present a mapping to DCOP for large-scale team coordination problems that were used in the DARPA Coordinators program.

This domain requires distributed, scalable algorithms to meet difficult bounds on computation and communication time. To achieve this goal, we develop a new DCOP algorithm that scales to problems involving hundreds of variables and constraints while converging to better solution qualities than existing DCOP algorithms. We show that our algorithm outperforms other DCOP algorithms for this domain and that our approach is competitive with other general approaches used in the DARPA Coordinators program.

CR Categories: I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence—Multiagent Systems;

Keywords: Distributed Constraint Optimization, Multi-agent Coordination, Task Scheduling

1 Introduction

Distributed Constraint Optimization (DCOP) is a general problem representation for multi-agent systems. Recent advances in DCOP algorithm development have led to an increasing number of application domains and focus on DCOP techniques. Recent applications of DCOP to real-world problems include sensor networks[Modi et al. 2005] and traffic flow cooperation [Junges and Bazzan 2008]. These existing problem domains for DCOP focus on small (<100 variables), deterministic domains. We present a mapping to DCOP for large-scale team coordination problems that were used in the DARPA Coordinators program, an algorithm to solve this mapping, and comparison of the results with other DCOP algorithms and DARPA team approaches.

2 C-TÆMS coordination problem

Multi-agent task planning and scheduling problems require a rich language for domain representation. The original TÆMS (Task Analysis, Environment Modeling, and Simulation) language was developed to provide a domain independent, quantitative representation of the complex coordination problem [Horling et al. 1999]. A C-TÆMS problem instance contains a set of agents and a hierarchically decomposed task structure. Nodes in the graph are either complex tasks (internal nodes) or primitive methods (leaf nodes). Each node may have temporal constraints on the earliest start time and the deadline. Nodes may also have non-local effect (NLE) constraints that represent hard (enables and disables) and soft (facilitates and hinders) node relationships. Methods have probabilistic outcomes for duration, quality, and cost. A sample C-TÆMS problem instance is shown in Figure 1.

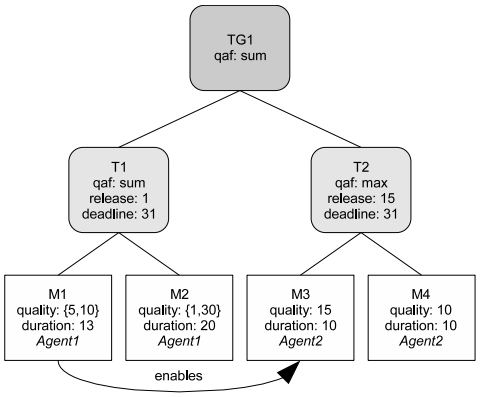

Figure 1: An example C-TÆMS problem instance.

3 Mapping C-TÆMS to DCOP

DCOP has been formalized in slightly different ways in recent literature, so we will adopt the definition as presented in [Petcu and Faltings 2005]. A Distributed Constraint Optimization Problem with n nodes and m constraints consists of the tuple $< X, D, U >$ where:

- $X = \{x_1,..,x_n\}$ is a set of variables, each one assigned to a unique agent
- $D = \{d_1,..,d_n\}$ is a set of finite domains for each variable
- $U = \{u_1,..,u_m\}$ is a set of utility functions such that each function involves a subset of variables in X and defines a utility for each combination of values among these variables (constraints)

An optimal solution to a DCOP instance consists of an assignment of values in D to X such that the sum of utilities in U is maximal. Our mapping for C-TÆMS to DCOP contains two parts: variable and constraint mappings.

Variables are created for each method and task in the C-TÆMS problem. Method variables are created with all possible start times as values and a value for *not scheduled*. Task variables can have several different sets of values depending on the type of QAF assigned to the task; these reflect whether a task is enabled for execution, if it accumulates quality, and if it is being forced to execute/not execute. In addition, a special end-time variable is created for each task with an outgoing NLE at or above it in the structure.

Constraints are created between each related node in the problem structure. Task-subtask and task-method constraints define how quality accumulation contributes to overall solution utility. Method-method mutex constraints en-

*e-mail: atlas@cis.udel.edu
†e-mail:decker@cis.udel.edu

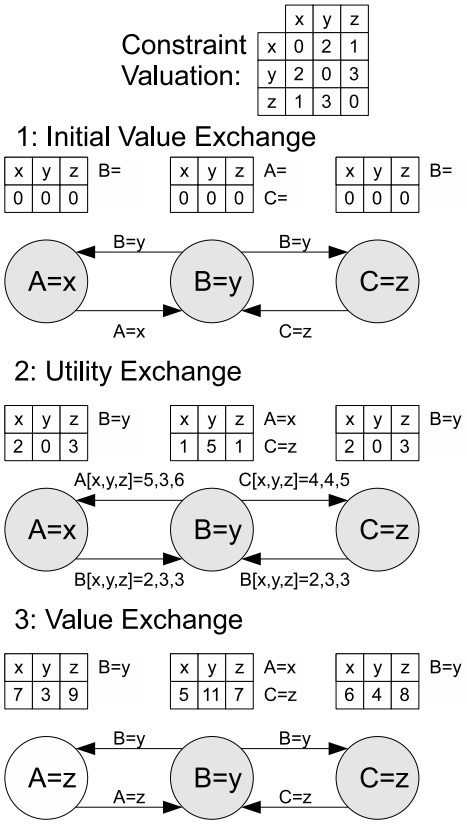

Figure 2: DNEA in action. Given constraint valuations shown at top and a random starting assignment of A=x, B=y, and C=z, DNEA finds the optimal assignment in one round of exchanges. Note how the utility exchange message from B to A in step 2 contains aggregated utility from C.

force that agents only perform one method at a time. NLE constraints between nodes enforce hard (enables and disables) and soft (facilitates and hinders) node relationships.

4 Distributed Neighbor Exchange

Our Distributed Neighbor Exchange Algorithm (DNEA) is similar in phases to other local value exchange based DCOP algorithms, including MGM, DSA/SCA, DBA, and max-sum. These algorithms exchange current variable assignments with neighbors, compute a maximization function based on neighboring assignments, and then choose to update or not update the local variable assignment. DNEA's main advantage is that it scales to large problems, operating in $O(\sum_{Y \in N_X} |D_Y| \cdot |D_X|)$ per cycle, while finding high-utility solutions using approximate neighborhood optimization. An example is shown in Figure 2.

5 Results

The DARPA COORDINATORs project uses a simulation framework based on simulated time ticks with dynamic execution outcomes. In Figure 3 we show comparison results between DNEA and other scalable local search algorithms (MGM2 and DSA2) using our C-TÆMS mapping in a static scenario. Results of our full solution on a subset of the real

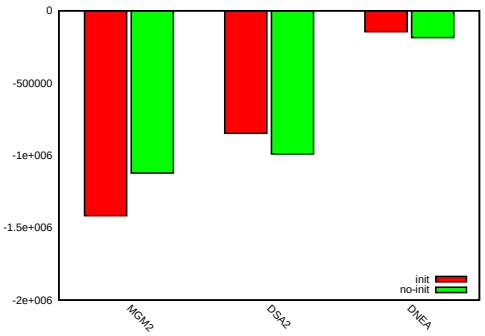

Figure 3: Estimated Solution Quality as a sum of DCOP utility for static scenario execution after 100 cycles, with and without an inital schedule. Closer to zero is better.

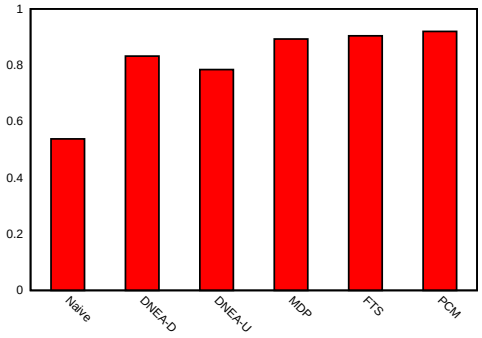

Figure 4: OptOP5PMix: Solution Quality as % of optimal (as determined by an offline solver).

DARPA test scenarios are shown in Figure 4. We show comparisons with a naive approach that follows a static schedule, and with the three contracted teams of the DARPA project (indicated by abbreviations of their approach).

These results are a significant step forward for DCOP techniques. Our mapping and algorithm successfully solves real-world problems orders of magnitude larger and more complex than previous DCOP applications. Our approach comes very close to the solutions of the DARPA teams, and shows promise for future improvements in this domain and applicability of DCOP to other real-world domains.

References

Horling, B. et. al., 1999. The TÆMS White Paper.

Junges, R., and Bazzan, A. L. C. 2008. Evaluating the performance of DCOP algorithms in a real world, dynamic problem. In *AAMAS '08*, 599–606.

Maheswaran, R. T. et. al. 2004. Taking DCOP to the real world: Efficient complete solutions for distributed multi-event scheduling. In *AAMAS '04*, 310–317.

Modi, P. et. al. 2005. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. In *AIJ*, 149–180.

Petcu, A., and Faltings, B. 2005. DPOP: A scalable method for multiagent constraint optimization. In *IJCAI 05*, 266–271.