Task Scheduling using Constraint Optimization with Uncertainty

James Atlas
Computer and Information Sciences
University of Delaware
Newark, DE 19716
atlas@cis.udel.edu

Keith Decker
Computer and Information Sciences
University of Delaware
Newark, DE 19716
decker@cis.udel.edu

1. INTRODUCTION

Multiagent task scheduling encompasses diverse domains of problems that require complex models and robust solutions. C-TÆMS [1] is a new specification, based on TÆMS [2], for multiagent task scheduling problems that represents the complex relationships necessary to model these domains.

Some recent work has been done in the area of mapping the C-TÆMS scheduling problem into a Distributed Constraint Optimization Problem (DCOP) [6]. Distributed constraint optimization is a direct extension to the traditional AI approach of constraint satisfaction for multi-valued constraints in a distributed system [7, 3]. Typical DCOP algorithms define the optimal solution as the optimal sum of local utilities.

Currently the mapping from C-TÆMS to a DCOP allows only for certain combinations of quality accumulation functions (QAFs), and works only for deterministic outcomes. The C-TÆMS scheduling problem contains uncertain information describing possible outcome distributions over the qualities of methods. The combination of these possible outcome distributions creates uncertainty in the global utility of a task schedule. Using an evaluation function for comparisons, the optimal schedule may not be equal to the one with the optimal sum of local utilities.

In this paper we extend the original DCOP formalization for uncertainty information in the form of utility distributions. Additionally, we extend the C-TÆMS mapping to include additional QAF combinations using only binary constraints. We then show how the C-TÆMS mappings can take advantage of the extended DCOP formalization with some sample evaluation functions. This research is ongoing, and comprehensive results on general classes of C-TÆMS scheduling problems are pending[1].

2. C-TÆMS SCHEDULING PROBLEM

The multiagent task planning and scheduling problem requires 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 [2]. 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 and soft precedence. Methods have probabilistic outcomes for duration, quality, and cost. Tasks have a quality accumulation function (QAF) that describes the quantitative combination of quality outcomes of subtasks and methods. Some basic QAFs include sum, min, max, and sync_sum. In the sample C-TÆMS problem instance in Figure 1, the node T1 represents a task with M1 and M2 representing a decomposition of this task into submethods. T1 has constraints for earliest start time of 1 and deadline of 31. The accumulated quality at T1 is a sum of the qualities of the executed submethods. In this case if both M1 and M2 executed within the temporal constraints, T1 would have an accumulated quality of $5 + 10 = 15$.

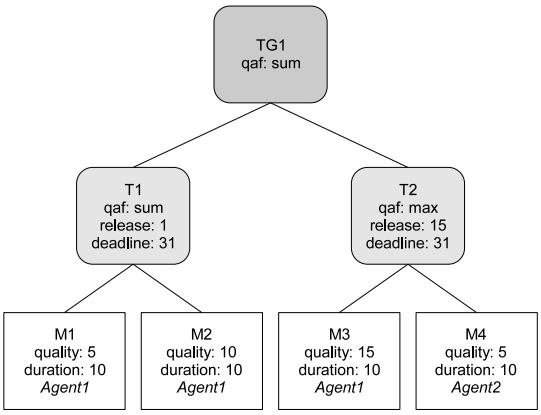

Figure 1: An example C-TÆMS problem instance.

3. DCOP FORMALIZATION

DCOP has been formalized in slightly different ways in recent literature, so we will adopt the definition as presented in [5]. 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

[1]This paper is intended as a positional paper. This is ongoing research and results are expected prior to AAMAS 07.

defines a utility for each combination of values among these variables

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. Problem domains that require minimum cost instead of maximum utility can map costs into negative utilities. The utility functions represent soft constraints but can also represent hard constraints by using arbitrarily large negative values.

4. EXISTING MAPPINGS FOR C-TÆMS SCHEDULING

A mapping for a subset of C-TÆMS to DCOP is proposed in [6]. The mapping using our formalization is:

- X = Each method is assigned to a unique variable.
- D = Unique domains for each variable containing all possible start times for the method assigned to the variable.
- U = Three types of utility functions:
  - Mutex constraints on all pairs of methods that share the same agent
  - For an NLE between two nodes, N_1 and N_2, all methods in the subtree of N_1 have a precedent constraint with all methods in the subtree of N_2
  - Unary soft constraints on each method that apply a cost if the method is not scheduled

This mapping allows only specific QAFs, enables NLE, and deterministic task outcomes. The complexity of the mapping, where M is the number of methods in the original C-TÆMS problem, involves O(M) variables, O(M^2) utility functions, and the size of each domain is O($|T|$) where T is the range of all possible start times.

5. PROPOSED MAPPINGS FOR C-TÆMS SCHEDULING

We can observe that DCOPs naturally optimize global sums of utility, so the mapping of the sum QAF can be achieved with relatively simple binary constraints. It is possible to map min and max QAFs using the existing mapping if they are the only QAF in a specified instance [6]. This is achieved by changing the DCOP aggregation function to a min or max function. The authors in [6] used ADOPT [4] to verify these mappings.

To date it has proven very difficult to combine QAFs in a hierarchical fashion. Although all types can be mapped using n-ary constraints over entire subtrees of tasks/methods, that increases the computational complexity. We propose the following mappings to binary constraints for non-sum QAFs that are correct for problem instances where non-sum QAFs apply only to methods and not to tasks or task groups.

5.1 sync_sum

This QAF produces quality equal to the sum of the methods that start at the same time slot. To start, we create a new variable that represents the synchronized start time of the methods involved in the sync_sum. This variable has the domain of all possible start times for any of the methods. Instead of the unary constraint producing a cost if the method is not scheduled, we create binary constraints between every method and the special sync_sum variable. If the method is not scheduled for the same time slot as the special sync_sum variable, the constraint returns a cost equal to the quality of that method.

5.2 min

This QAF produces quality equal to the minimum quality of any submethod. An important note is that the min QAF produces no quality if any of the methods are not scheduled. To map this QAF, we again create a special variable. This variable has the domain of true and false. We first determine which of the submethods has the lowest potential quality. A special binary constraint is created between this method, a, and the special variable, v. The cost function for this constraint is:

$$cost(a) = \begin{cases} a_{qual} & \text{if } v = false \\ \infty & \text{if } v = true \text{ and } a \text{ not scheduled} \\ 0 & \text{otherwise} \end{cases}$$

Next, we create a binary constraint between each other method, b, and the special variable, v, with the cost function:

$$cost(b) = \begin{cases} \infty & \text{if } v = true \text{ and } b \text{ not scheduled} \\ 0 & \text{otherwise} \end{cases}$$

5.3 max

This QAF produces quality equal to the maximum quality of any submethod. To map this QAF, we create another special variable. This variable's domain values are the set of all possible qualities produced by any submethod (and a special tag that marks that quality's submethod) and a not scheduled value. We first determine which of the submethods has the highest potential quality, q_{max}. We create a binary constraint between each of the methods, c and the special variable, x. If the method, c, is scheduled we use the cost function:

$$cost(c) = \begin{cases} \infty & \text{if } c_{qual} > x \\ q_{max} - c_{qual} & \text{if } c_{qual} = x \text{ and } x_{tag} = c \\ 0 & \text{otherwise} \end{cases}$$

If the method, c, is not scheduled we use the cost function:

$$cost(c) = \begin{cases} 0 & \text{if } x_{tag} \neq c \\ q_{max} & \text{if } x \text{ not scheduled and } q_{max} = c_{qual} \\ \infty & \text{otherwise} \end{cases}$$

6. UNCERTAINTY IN THE C-TÆMS SCHEDULING PROBLEM

Uncertainty of various task characteristics, such as completion time, solution quality, and total cost, is one of the major complexities underlying the C-TÆMS scheduling problem. In our previous discussion of mappings, we do not attempt to include this concept. All the mappings so far discussed are based on deterministic outcomes.

Uncertainty can be represented in an agent system as a statistical distribution of values. We may incorporate a crude mapping of this uncertainty into the existing DCOP

framework by using average expected value for the uncertainty (a sum product of the probabilities and values). However, this offers little help in domains where the global utility over the uncertainty is not an average expected value. For many domains including C-TÆMS scheduling, it is valuable to incorporate risk aversion functions into the global utility, or enforce minimum confidence level utility.

6.1 DCOP with Utility Distributions

We can extend the DCOP problem formulation to include uncertainty by allowing constraint evaluation functions to return a distribution instead of a single value. A global optimum is now an optimal distribution instead of a maximum (or minimum) sum. To evaluate the optimality of a distribution, evaluation criteria must be formalized. The optimal evaluation function may not be the same for all problems for all agents. Thus we must include the evaluation function as part of the extended DCOP problem. We extend our previous DCOP formalization for this:

- $U = \{u_1,..,u_m\}$ is a set of utility functions such that each function involves a subset of variables in X and defines a utility distribution for each combination of values among these variables
- $u = \{(u_p^1,\ u_v^1),..,(u_p^t,\ u_v^t)\ \}$ is a distribution of probabilities and values such that $\sum_{r=1}^{t} u_p^r = 1$
- $E = \{e_1,..,e_n\}$ is a set of evaluation functions for each variable that reduce a utility distribution to a single utility value; $e(u) = v$ where v is a single utility value

6.2 Re-mapping C-TÆMS to DCOP

The C-TÆMS scheduling problem mapping presented in the previous section can easily be extended to include this concept of uncertainty. We maintain the current mappings for all of the hard constraints (ones that return either zero or infinite cost). For each soft constraint we return a set of utility distributions instead of the single value. This allows a method to specify that it produces a distribution of quality; for example in Figure 1 method M1 may now produce quality = 5 for 80% of executions, quality = 10 for 10% of executions, and quality = 100 for 10% of executions. This would be represented as a utility distribution of {(0.8, 5), (0.1, 10), (0.1, 100)}. Also, for the C-TÆMS scheduling problem we use a single evaluation function, E, for all agents.

We illustrate three sample evaluation functions among many that express the effectiveness of our model. The first function is a risk neutral expected value function that simply computes a sum product of the utility distribution. For the prior example distribution, this function would evaluate as:

$$e(u) = 0.8 \cdot 5 + 0.1 \cdot 10 + 0.1 \cdot 100 = 15$$

The second is a risk averse evaluation function based on quadratically decreasing utility, where:

$$e(u) = (\sum_{r=1}^{t} u_p^r \cdot \sqrt{u_v^r})^2$$

For the example this would evaluate as:

$$e(u) = (0.8 \cdot \sqrt{5} + 0.1 \cdot \sqrt{10} + 0.1 \cdot \sqrt{100})^2 = 9.64$$

The third is a minimum confidence evaluation function, such that utility equals the highest value v such that c% of the values are greater than or equal to v. For the example this would evaluate as 100 for $c = 5$%, 10 for $c = 20$%, and 5 for $c = 90$%. Many other evaluation functions are available, but these simple functions will allow our global solution to incorporate things such as risk aversion and minimum confidence.

It is straightforward to aggregate utility distributions using the typical summation function. Thus a global aggregated distribution for a specific variable assignment is possible. Applying the evaluation function to the aggregated distribution allows for a choice of an optimal variable assignment. However, many DCOP algorithms also require intermediate comparisons between the values, which are distributions in our model. Appropriate comparison mechanisms that correctly identify the optimal global assignment are a topic of current research. This research and implementation is ongoing, and comprehensive results on general classes of C-TÆMS scheduling problems are pending.

7. CONCLUSION AND FUTURE WORK

We have introduced a new formalization for DCOP that includes uncertainty characteristics. We showed how a problem domain that includes uncertain outcomes, the C-TÆMS scheduling problem, can be mapped into our new formalization. We also introduced some additional mappings for C-TÆMS QAFs to binary constraints. Using our new formalization we illustrated how evaluation functions can express concepts such as global risk aversion and minimum confidence.

We are currently testing our formalization in several domains to see how easily it can be applied to various problems that involve uncertainty. Additionally we plan to develop comparison tests to calculate the effectiveness of the model. We will also continue to improve the mapping of the C-TÆMS scheduling problem to the DCOP formalization.

8. REFERENCES

[1] M. Boddy, B. Horling, J. Phelps, R. P. Goldman, and R. Vincent. C-tæms language specification, 2005.

[2] B. Horling, V. Lesser, R. Vincent, T. Wagner, A. Raja, S. Zhang, K. Decker, and A. Garvey. The TAEMS White Paper, January 1999.

[3] J. Liu and K. P. Sycara. Exploiting problem structure for distributed constraint optimization. In *Proceedings of the First International Conference on Multi–Agent Systems*, pages 246–254, 1995.

[4] P. Modi, W. Shen, M. Tambe, and M. Yokoo. Adopt: Asynchronous distributed constraint optimization with quality guarantees. *AIJ*, pages 149–180, 2005.

[5] A. Petcu and B. Faltings. Dpop: A scalable method for multiagent constraint optimization. In *IJCAI 05*, pages 266–271, Aug 2005.

[6] E. Sultanik, P. J. Modi, and W. Regli. On modeling multiagent task scheduling as a distributed constraint optimization problem. In *IJCAI*, 2007.

[7] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara. Distributed constraint satisfaction for formalizing distributed problem solving. In *International Conference on Distributed Computing Systems*, pages 614–621, 1992.