Solving the Multiagent Selection and Scheduling Problem

James Boerkoel
Computer Science and Engineering
University of Michigan, Ann Arbor
boerkoel@umich.edu

Introduction

Consider Ann's morning scheduling problem. Ann is a graduate student, who, among many other objectives, would like to both exercise and work on research during her morning. I summarize possible morning activities in Table 1. Even for these two simple objectives, selecting a feasible schedule from the many possible schedules (run then generate experimental results, write then bike, swim then read some related work, etc.) may be non-trivial. However, suppose Ann wants to run with her friend Bill, who notoriously oversleeps his alarm. Additionally, suppose also that Ann must coordinate the use of her lab's computational cluster with her lab mate Claire.

Without further information from Bill and Claire, it is impossible for Ann to determine which candidate schedules will successfully achieve her morning goals. One option for Ann would be to myopically select her schedule anyway, with the risk that her attempt to run with Bill or to use the computational cluster could result in a failed goal. As another option, Ann could also volunteer to collect the scheduling constraints of both Bill and Claire and generate a single joint morning schedule. However, this puts additional scheduling burden on Ann while requiring both Bill and Claire to reveal other scheduling commitments they may prefer to keep private. Even if Ann employed a centralized computational agent to solve this global scheduling problem, the resulting combinatorics may limit the scalability of such a centralized approach. Instead, the pervasiveness of personal computational devices, coupled with desires for scalability and privacy, argue for decentrally solving such problems using multiagent algorithms.

My thesis focuses on providing scalable, multiagent algorithms for solving rich, complex multiagent activity scheduling *and* selection problems, while retaining as much privacy as possible on behalf of the human users. My approach is distinct from other recent multiagent scheduling approaches (Hunsberger 2002; Smith et al. 2007; Shah, Conrad, and Williams 2009) in that it uses a constraint-based representation of *selection* (finite-domain) aspects of scheduling problems in addition to the scheduling aspects. I proceed by introducing the Multiagent Selection and Scheduling Problem (MASSP), explaining my approach for solving the MASSP and methodology for evaluating my approach, presenting preliminary results, and concluding with expected contributions.

Activity	Objective	Duration	External Constraints
Run	Exercise	[30,45]	Synch w/ Bill
Swim	Exercise	[45,75]	Pool Hours
Bike	Exercise	[60,90]	-
Experiment	Research	[120,180]	Mutex w/ Claire
Read	Research	[90,120]	-
Write	Research	[180,240]	-

Table 1: Summary of Ann's morning activities.

Problem Statement

In this section, I introduce a novel problem representation, the Multiagent Selection and Scheduling Problem (MASSP), which generalizes Schwartz's (2007) Hybrid Scheduling Problem (HSP) representation to a multiagent setting. Ann's local problem can be represented as an HSP, depicted graphically in Figure 1. As background, the HSP formulation combines the typical finite-domain Constraint Satisfaction Problem (CSP) formulation with the Disjunctive Temporal Problem (DTP) formulation and can be used to represent Ann's scheduling problem. An HSP is defined as the tuple $\langle V, C \rangle$. The set of variables V is partitioned into two sets. V_F is a set of finite-domain variables (unshaded nodes in Figure 1), each of which has an associated domain with a finite number of possible values. In the current example, Ann's HSP might include finite-domain variables E (exercise) and R (research) with domains $\{run, swim, bike\}$ and $\{write, read, experiment\}$ (represented by unary constraints) respectively. V_T is a set of timepoint variables (shaded nodes in Figure 1), each of which represents an event and has an implicit, continuous domain of times. Ann may specify timepoint variables E_{ST}, E_{ET}, R_{ST}, and R_{ET} representing the start and end times of her exercise and research activities. These timepoints are grounded against a special zero reference timepoint variable (i.e. clock).

The set of constraints C is partitioned into three sets. C_F is the set of finite-domain constraints, each of which is repre-

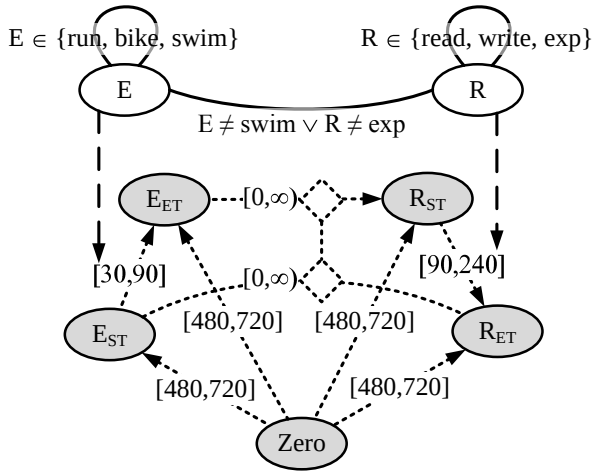

Figure 1: An HSP representation Ann's morning schedule

sented in Figure 1 as a solid edge. Each finite-domain constraint is defined over some subset of V_F and specifies the allowable (or alternatively impermissible) combinations of values that can be assigned to the variables in the subset. For example, Ann may prohibit herself from both swimming and conducting experiments ($E \neq swim \vee R \neq experiment$), since the pool and research lab are in opposite corners of campus compared to her centrally-located office.

C_T is the set of disjunctive temporal constraints, represented in Figure 1 as dotted edges. A disjunctive temporal constraint is defined over timepoint variables, and is satisfied when at least one of a disjunctive set of possible temporal difference constraints is satisfied. A temporal difference constraint is a constraint of the form ($v_i - v_j \in [-B_{ji}, B_{ij}]$), where $v_i, v_j \in V_T$ and B_{ij} ($\geq v_i - v_j$) and B_{ji} ($\geq v_j - v_i$) are bounds on the difference between v_i and v_j. For example, the constraint ($E_{ST} - R_{ET} \in [0, \infty] \vee R_{ST} - E_{ET} \in [0, \infty]$) represents that Ann cannot exercise and research at the same time. I represent disjunction graphically in Figure 1 by appending each temporal difference disjunct with a diamond, connecting all disjuncts that belong to the same disjunctive constraint. Together, the temporal variables and constraints form a DTP, $\langle V_T, C_T \rangle$. An important, special case of the DTP is the Simple Temporal Problem (STP), which is a DTP where all the disjunctive constraints in C_T contain only a single disjunct. The STP is important since it is polynomial-time solvable and often solved as a subproblem of more complex scheduling problems.

Finally, C_H is the set of hybrid constraints, represented in Figure 1 as dashed edges. A hybrid constraint is composed of a finite-domain constraint and a disjunctive temporal constraint, and is satisfied when at least one of its associated constraints is satisfied. For example, ($E = run \rightarrow E_{ET} - E_{ST} \in [30, 45]$) represents, in implicative form, that Ann will exercise for between 30 and 45 minutes if she chooses to run. A solution schedule for an HSP is a complete assignment of values to variables that is consistent with all constraints.

Having reviewed the HSP formulation, a MASSP is easily defined as the tuple $\langle A, C \rangle$. A is the set of agent problems, $\{A_1, \ldots, A_i, \ldots, A_n\}$ for the n agents, where an agent problem, A_i, is specified by an HSP, $\langle V^i, C^i \rangle$. C is the set of interagent constraints where each constraint in C is specified over variables from at least two different agents. For example, C would include the interagent, hybrid constraint that dictates if both Ann and Claire plan to use the computing cluster, they must do it at mutually exclusive times. A solution to the MASSP is composed of solutions to each individual agent problem that jointly respect the interagent constraints. Further we are interested in algorithms that solve this problem without revealing *private timepoints*, timepoints involved in no interagent constraints, or *private constraints*, constraints involving at least one private variable.

Approach

In this section, I describe how I adapt the typical CSP-based approach of interleaving consistency maintenance with variable assignment to solve the MASSP.

Implicit Constraint Elucidation

The MASSP is defined as the composition of agent HSP subproblems, coupled by a set of interagent constraints. Similarly, each agent HSP subproblem is composed of a CSP and DTP subproblem, connected by hybrid constraints. Thus, a MASSP is naturally decomposable into many single-agent CSPs and DTPs. There exist many highly efficient variable assignment heuristics and consistency maintenance techniques for solving single-agent CSPs and DTPs. To take advantage of these techniques in my MASSP solution algorithms requires effectively communicating implications of constraints between subproblems. Using Ann's morning scheduling problem as an example, as Ann selects the activities she would like to schedule (Ann's finite-domain CSP), she should consider if she can consistently schedule the activities she is selecting (Ann's DTP) and *vice-versa*. Similarly, as Ann solves her morning scheduling problem, she should take account how her decisions will affect Bill and Claire's scheduling decisions, and *vice-versa*.

The success that maintaining high levels of consistency between variable assignments has had for solving single-agent CSPs and DTPs argues for taking a similar approach for solving the MASSP. However, since techniques for maintaining high levels of consistency *within* single-agent CSPs and DTPs are relatively well established, I propose a process that I call Implicit Constraint Elucidation (ICE) for increasing consistency *between* the subproblems of the MASSP. I hypothesize that agents can reduce overall solution time by orders of magnitude when applying ICE in conjunction with off-the-shelf, single-agent consistency maintenance techniques.

ICE encourages an agent to spend time introspectively quantifying how the internal constraints of its subproblems affect other subproblems and then elucidating these effects as additional explicit constraints over the other subproblems. This approach is similar to marginalization in Bayesian learning algorithms, where instead of calculating

a prior probability given all possible hypotheses, we elicit inevitable constraints on one subproblem given all possible variable assignments to another subproblem. This approach has many advantages. First, by exchanging explicit constraints, two agents sharing an interagent constraint may realize that these explicit constraints render the original interagent constraint superfluous, and thus decouple their subproblems from each other. Second, since an agent receives explicit constraints from other agents expressed in terms of its own subproblem, it can use these constraints to prune its search space, inform its heuristics, and mitigate the chances of making a globally infeasible variable assignment.

A third advantage of the ICE approach is a key one: *privacy*. As mentioned before, an agent can naturally partition its variables into a *private* subset, those involved in no interagent constraints, and a *shared* subset, those involved in at least one interagent constraint. An agent using ICE can proactively 'marginalize' away all information about private variables *before* communicating with another agent. So an agent using ICE never needs to reveal any information about its private variables or any constraints specified over one or more of its private variables. Instead, agent messages consist of constraints composed entirely of shared variables. For any given MASSP instance, these constraints are sufficient for determining global MASSP consistency. So if a person wishes to keep aspects of her schedule private, she can do so naturally by not involving them in any interagent constraints.

Guiding Search in the MASSP

Consistency maintenance prunes provably infeasible values from variable domains. Because of this, agents can asynchronously execute MASSP consistency maintenance techniques without concerns about eliminating sound schedules or proposing unsound schedules. When an agent assigns a value to a variable, on the other hand, it does so without guarantees that such an assignment will lead to a sound schedule or that it will not eliminate sound schedules. In single-agent CSP search algorithms, this is addressed through a backtracking search that systematically guarantees that the entire search space is explored. However, if agents make such variable assignments asynchronously, this task becomes much more challenging, though such an approach is the foundation of many distributed CSP solution techniques (Yokoo et al. 1998).

An alternative to an asynchronous backtracking search is to introduce a decoupling. A decoupling is a set of local agent constraints that make the set of interagent constraints superfluous. As a special case, Hunsberger (2002) defined the temporal decoupling problem for an STP instance distributed across multiple agents. In terms of Ann's morning scheduling problem, Ann and Claire's schedules are necessarily coupled by their use of a common resource. However, suppose that Claire agrees to complete her use of the computational cluster by 10:00 AM (a local constraint) and that Ann agrees to wait to begin her use of the cluster until 10:00 AM (another local constraint). At this point, the interagent constraint preventing Ann and Claire's use of the cluster from overlapping becomes unnecessary, since the two local constraints already imply that Ann and Claire will never use the cluster at the same time. Further, once Ann and Claire agree to these local constraints, Ann's can proceed to solve the remainder of her scheduling problem without concern for how her decision will affect Claire. So, once all interagent constraints have been decoupled, agents can perform search over their local problems in an asynchronous manner.

Unfortunately, Planken, de Weerdt, and Witteveen (2010) prove that finding such a decoupling is, in general, as hard as solving the underlying problem. However, I believe that for loosely coupled problems, agents can exploit both asynchronous consistency maintenance techniques and concurrent agent search to quickly evaluate many such decouplings. This reduces the problem into two stages: 1) finding a decoupling and 2) solving local agent problems in response to this decoupling. The rest of this section focuses on techniques for finding a decoupling, since techniques for efficiently solving single-agent HSPs already exist (Schwartz 2007; Boerkoel and Durfee 2008).

I hypothesize that finding a MASSP decoupling is best done incrementally, by interleaving the decoupling of individual interagent constraints with the MASSP consistency maintenance techniques described in the previous section. Finding a decoupling incrementally allows inconsistent decouplings to be pruned prior to committing to a complete decoupling. Next, I will evaluate heuristics for deciding which interagent constraint to decouple next. Candidate heuristics include an adaptation of graph partitioning algorithms for selecting decoupling points that lead to the greatest amount of partitioning in the agent network. Alternatively, I can adapt *most-constrained variable* style heuristics to identify the most *coupled* variables. Similarly, I can adapt *least-constrained value* style heuristics for identifying which decoupling constraints will least reduce search space of agent subproblems.

Methodology

The distributed CSP literature defines a variety of metrics for evaluating the efficiency of solution algorithms. Two of the most widely accepted metrics are consumed communication bandwidth and nonconcurrent constraint checks. These two metrics quantify two major costs of real multiagent systems: the amount of necessary communication and how well agents can exploit computational concurrency. Since agents in real-world multiagent systems may have differing communication and computational abilities, these metrics allow neutral comparison across distinct multiagent systems and for experimentation in simulation. I will evaluate my hypotheses by comparing the performance of my techniques for solving MASSP to current centralized solvers as well as more naïve versions of MASSP search using these metrics.

Preliminary Results

In this section, I briefly discuss a few of my preliminary results and their roles in my dissertation.

Hybrid Constraint Tightening

My ICE-inspired Hybrid Constraint Tightening (HCT) preprocessing algorithm (Boerkoel and Durfee 2008; 2009) reformulates hybrid constraints by lifting information from

the structure of an HSP instance. These reformulated constraints elucidate implied constraints between the CSP and DTP subproblems of an HSP earlier in the search process, leading to significant search space pruning. Despite the computational costs associated with applying the HCT preprocessing algorithm, HCT leads to orders of magnitude speedup when used in conjunction with off-the-shelf, state-of-the-art solvers, as compared to solving the same problem instance without applying HCT. Additionally, I have explored properties of HSPs that influence HCT's efficacy, quantifying the influence empirically. I believe that HCT will continue to play an important role for solving the MASSP, since an agent that can more quickly understand the temporal implications of a selection decision (and *vice-versa*) can use this information to more quickly express new, precise, and otherwise implicit interagent constraints.

The Multiagent Simple Temporal Problem

Another positive result is my extension of STP solution techniques to multiagent problems (Boerkoel and Durfee 2010). My multiagent STP algorithms achieve impressive speedup over centralized approaches, especially as problems become more weakly coupled. This is achieved by partitioning the STP into private components for each agent and a single shared STP component. Then, agents are able to work on their private components independently and concurrently, using ICE principles to quantify how their local subproblem affects other agents. This partitioning leads to important privacy and computational concurrency gains. Solving the multiagent STP is an important precursor for solving the multiagent versions of more complicated scheduling formulations such as DTPs and HSPs because many solution methods for solving the DTP and HSP involve evaluating partial and candidate assignments that form *component STPs*, which need to be quickly evaluated for consistency (Tsamardinos and Pollack 2003).

Preference-based Decoupled Constrained Optimization with CP-nets

Conditional preference networks (CP-nets) are models of utility that allow people to naturally express qualitative preferences using *ceteris paribus* semantics (Boutilier et al. 2004). Using CP-nets allows me to find a solution that not only satisfies the constraints, but also is optimal relative to the CP-net. I demonstrate (Boerkoel, Durfee, and Purrington 2010) that CSP-centric variable ordering heuristics can be replaced with CP-net inspired heuristics. This leads to an interleaving of preference with constraint reasoning that dominates other approaches, which either allow the CSP reasoning to make additional unsolicited scheduling decisions based on computational efficiency or which allow the user to exert multiple preferences before refining the search space.

Expected Contributions

To summarize, in addition to the contributions mentioned in the preliminary results, I expect to contribute the following: 1) a constraint-based formulation for representing multiagent problems involving both activity selection and scheduling; 2) an algorithm, and evaluation thereof, for maintaining consistency of multiagent CSPs, multiagent DTPs, and the MASSP using ICE principles; 3) a decoupling strategy for multiagent CSPs, multiagent DTPs, and the MASSP; and finally 4) an evaluation of an algorithm combining all techniques in simulated multiagent environments.

Acknowledgments

I thank the anonymous reviewers and my advisor, Ed Durfee, for their comments and suggestions. This work was supported, in part, by the NSF under grant IIS-0534280 and by the AFOSR under Contract No. FA9550-07-1-0262.

References

Boerkoel, J., and Durfee, E. 2008. Hybrid Constraint Tightening for Hybrid Constraint Scheduling. In *Proc. of AAAI 2008*, 1446–1449.

Boerkoel, J., and Durfee, E. 2009. Evaluating hybrid constraint tightening for scheduling agents. In *Proc. of AAMAS 2009*, 673–680.

Boerkoel, J., and Durfee, E. 2010. Algorithms for Solving the Multiagent Simple Temporal Problem. In *Proc. of ICAPS 2010*, To Appear.

Boerkoel, J.; Durfee, E.; and Purrington, K. 2010. Generalized Solution Techniques for Preference-Based Constraint Optimization with CP-nets. In *Proc. of AAMAS 2010*, To Appear.

Boutilier, C.; Brafman, R.; Domshlak, C.; Hoos, H.; and Poole, D. 2004. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *JAIR* 21:135–191.

Hunsberger, L. 2002. Algorithms for a temporal decoupling problem in multi-agent planning. In *Proc. of AAAI 2002*, 468–475.

Planken, L.; de Weerdt, M.; and Witteveen, C. 2010. Optimal Temporal Decoupling in Multiagent Systems. In *Proc. of AAMAS 2010*, To Appear.

Schwartz, P. 2007. *Managing complex scheduling problems with dynamic and hybrid constraints*. Ph.D. Dissertation, University of Michigan.

Shah, J.; Conrad, P.; and Williams, B. 2009. Fast distributed multi-agent plan execution with dynamic task assignment and scheduling. In *Proc. of ICAPS 2009*, 289–296.

Smith, S.; Gallagher, A.; Zimmerman, T.; Barbulescu, L.; and Rubinstein, Z. 2007. Distributed management of flexible times schedules. In *Proc. of AAMAS 2007*, 472–479.

Tsamardinos, I., and Pollack, M. 2003. Efficient solution techniques for disjunctive temporal reasoning problems. *Artificial Intelligence* 151(1-2):43–89.

Yokoo, M.; Durfee, E.; Ishida, T.; and Kuwabara, K. 1998. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE TKDE* 10(5):673–685.