Plan Execution by Contracting in a Multi-Agent Environment

John Collins, Maxsim Tsvetovat and Maria Gini
Dept of Computer Science and Engineering, University of Minnesota

Bamshad Mobasher
School of Computer Science, Telecomm., and Info. Systems, DePaul University

Abstract

We present a computational model of planning, scheduling, and execution that is capable of supporting the interactions of a self-interested agent with other agents in a contracting environment over an extended period of time. The model is appropriate for situations in which a customer agent, in order to fulfill its goals, must contract with other supplier agents for all or part of the necessary tasks. The agents are assumed to be self-interested and with limited rationality. Supplier agents attempt to gain the greatest possible benefit, and customer agents will attempt to pay the lowest price.

1 Introduction

The University of Minnesota's MAGNET (Multi-Agent Negotiation Testbed) system is an innovative agent-based approach to complex logistics and supply-chain management problems. It is designed to support the execution of complex plans among a population of independent, autonomous, heterogeneous, self-interested agents. We call this activity *Plan Execution By Contracting*. The MAGNET system is based on three elements:

1. The *agents*. Each agent is an independent entity, with its own structure, goals, and resources. In general, the resources under "control" of an individual agent are not sufficient to satisfy that agent's goals, and so the agent must negotiate with other agents in its environment in order to meet its goals.

2. The *market*. Agents find each other and carry out negotiations through a distributed infrastructure that enforces protocol rules, limits opportunities for fraud and counter-speculation, and tracks the requests, commitments, and progress toward toward goals among the agent population.

3. The *protocol*. Although the basic structure of the agents and the market will support a variety of negotiation protocols, we have developed a finite, leveled commitment protocol that limits the time and bandwidth required for the negotiation process without limiting the scope of possible agreements.

Because it is based on a market-based economic model, a MAGNET system acts to allocate resources to their highest-value uses over time in a completely distributed fashion. Because MAGNET agents are heterogeneous and self-interested, they may represent real-world entities who may tune their levels of cooperation and competitiveness to suit their own needs.

2 A MAGNET Agent

In general, a MAGNET agent has four basic functions: planning, negotiation, execution monitoring, and resource management. Within the scope of a negotiation, we distinguish between two agent *roles*, the *Contractor* and the *Supplier*. A Contractor is an agent who has a plan to satisfy some goal, and needs resources outside its direct control in order to carry out that plan. The plan may have a *value* that varies over time. In response to a Call For Bids, some set of Supplier agents may offer to provide the requested resources or services, for specified prices, over specified time periods. Figure 1 shows the general relationships among Contractor agents, Supplier agents, and the Market.

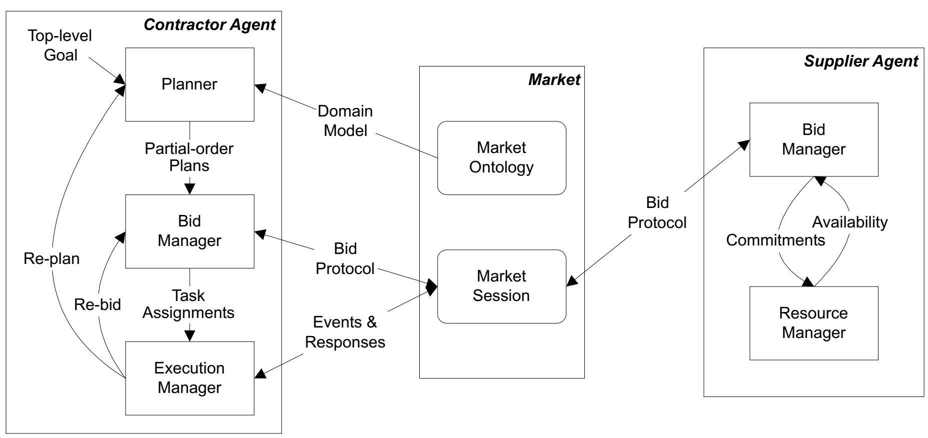

Figure 1: MAGNET Agents

Once the Contractor agent receives bids, it must evaluate them based on cost, risk, and time constraints, and select the optimal set of bids (or parts thereof) which can satisfy its goals [3]. This *task assignment* forms the basis of an initial schedule for the execution of the tasks.

After a set of bids are selected, Suppliers are notified of their commitments, and the Contractor agent invokes its Execution Manager to oversee completion of the plan. Plan maintenance behaviors include re-negotiating existing commitments, re-bidding portions of the plan, re-planning for subgoals that are in jeapordy, and abandoning the goal.

3 The MAGNET Market Infrastructure

The MAGNET system incorporates an explicit market infrastructure. This is a distributed, hierarchical system that manages security, tracks commitments and performance of agents, and logs their interactions. The market specifies the terms of discourse among agents through an *Ontology*, and keeps historical data that may be used by agents in their risk evaluations. Different market segments may specify specialized ontologies. For example, one market segment may cover transportation services, and another construction services. Agents who wish to offer resources and services do so through one or more market segments whose ontologies describe their offerings.

Within the MAGNET market framework, negotiation activities are encapsulated in a *market session* [4]. A market session (or simply a session) is the vehicle through which market services are delivered dynamically to participating agents. It serves as an encapsulation for a transaction in the market, as well as a persistent repository for the current state of the transaction. We have chosen the term "session" to emphasize the temporally extended nature of many of these interactions. For example, if an agent wishes to build a fuel depot, it initiates a session and issues a call-for-bids. The session extends from the initial call-for-bids through the negotiation, awards, construction work, and the final settlement. In other words, the session encloses the full life of a contract or a set of related contracts. The session mechanism ensures continuity of partially-completed transactions, and relieves the participating agents from having to keep track of detailed negotiation status themselves.

The MAGNET negotiation protocol consists of a *contracting* phase and a *execution* phase. The contracting phase is a simple three-step process consisting of a Call For Bids, a Bidding cycle, and an Award cycle. We avoid the need for open-ended negotiation by means of bid break-downs and time-based decommitment penalties. The execution phase may involve negotiations over schedule adjustments, decommitments, and in some cases repititions of the bidding cycle when it becomes necessary to re-allocate resources that had originally been committed.

The interactions involved in the basic bidding and execution cycle among the contractor, supplier, and market session are illustrated in Figure 2. During execution, the interactions can be much more complex than indicated here, since either party may decommit from a contract (and pay a penalty), and the contractor is continuously monitoring and repairing its plan by replanning and rebidding when events fail to proceed according to expectations.

4 The Market Ontology

To facilitate the negotiation process and provide a common representation of objects and concepts among agents, a MAGNET market provides a *market ontology* as a unified contract specification system. The roles of the contract specification systems are:

- *Descriptive:* to provide a description of a process that has already occurred;
- *Prescriptive:* to provide a recipe describing how a process can occur;
- *Semantic:* to provide a semantic model, define objects and activities and establish the scope of the system;
- *Communicative:* to enable interoperability between different agents in the marketplace.

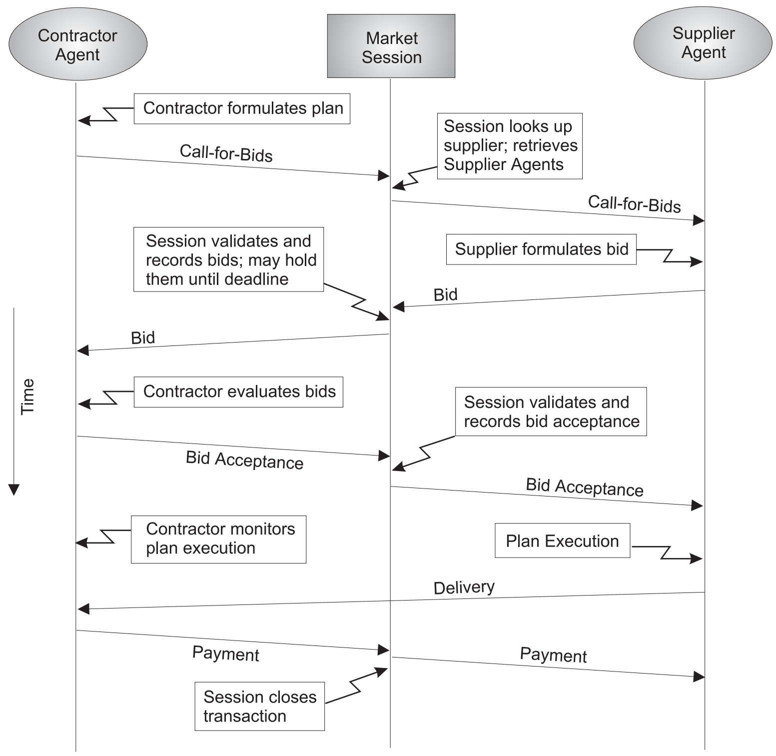

Figure 2: A Typical Session-Mediated Negotiation

The contract specification system is composed of several classes of elements: *Atoms* are the basic units of trade in the system. An atom can represent a physical object, a unit of a resource, or a unit of work done. A market ontology contains a fixed set of atoms which define the terms of discourse in a particular market. There are several main types of atoms, characterized by their behavior in the systemi, such as abstract, concrete, consumable, capital, whole, fractional, etc.

Actions represent all activities that are possible in the system. An action is defined as $A = \{P, [T], [S]\}$ where P is a set of input and output ports that define the required materials and tools as well as the result of the action, T is an optional set of time constraints, and S is the state.

A port specifies the type of atom that is required to commence the action or the type of atom produced as the result of this action. There are several types of ports that define what happens to an atom that has been passed through a given port.

Consume is an input port that signifies a resource to be used up in process of completing an action. The atom passing through a Consume port will not exist after the action has started. *Produce* is an output port that creates an atom which did not exist before the performance of the action. *Use* represents a resource that is exclusively used for performance of the action, and released once the action has concluded. Two other types of ports, *Specify* and *ToSpec*, allow for dynamic creation of atoms in the system. *Specify* produces an abstract atom specifying the action to occur at some later time, and *ToSpec* performs an action specified by such abstract atom. Within a particular market, actions defined in the ontology represent the production capabilities of actual or potential suppliers in that market.

There are a number of issues that have to be resolved in order for the ontology to be useful in the real world. *Authentication issues* arise when atoms in an electronic commerce system must represent real physical goods. Every physical object is unique in the sense that one cannot make a perfect atom-by-atom copy of it (and even if that was possible, it won't be made out of the same atoms). An electronic representation of such physical object must also be unique, even though data objects can be freely copied. One way to achieve that uniqueness is to produce physical objects that carry a serial number, and include an encrypted version of the same serial number in its electronic representation.

Money is just one type of a fractional atom. Thus, nothing prevents some market from declaring its universal medium of exchange as something else (shells, colored beads or crude oil). In fact, nothing prevents the economy from developing multiple and parallel mediums of exchange, or from building a barter-based economy. This makes bidding and evaluation of bids a much more complex process, and therefore agents or markets may need to specify the types of "currency" allowed in exchanges.

5 Components of the Contractor Agent

5.1 Planner

A plan is a combination of operations to be executed by the agent itself and operations that are available from other agents, as expressed in the Market Ontology. The planner need not linearize the plans it emits with respect to time, since multiple agents in the market environment are capable of executing operations in parallel. Instead, temporal constraints among plan operations are embedded in the plan and included in the call-for-bids.

The plan provided to the bid manager consists of a set of task descriptions of nonzero length, the temporal constraints among them, and possibly nonzero delays between tasks, to cover communication and transportation delays. The agent incorporates a temporal reasoning system similar to Dean and McDermott's Time Map Manager [5] to build and maintain this representation.

Our decision to use a traditional non-linear planner, as opposed to a formulation that integrates planning and scheduling [11], is based on the fact that the integrated approach operates from a resource-based time line, and in our system the set of resources is unknown at the time the planner operates.

5.2 Bid Manager

The bid manager is responsible for ensuring that resources are assigned to each of the tasks of a plan, that the assignments taken together form a feasible schedule, and that the cost of executing the plan is minimized. This cost must also be less than the value of the goal at the time the goal is reached. When the bid manager is invoked, some tasks in the plan may already be assigned. This is because the contract supervisor may use the bid manager to repair a partially-completed plan in which previously determined assignments have failed.

The bid manager must perform these tasks in order to carry out its responsibilities:

1. Construct and Issue Call For Bids. The call-for-bids contains a subset of the tasks in the plan with their precedence relations. There might be elements that are not included in the call-for-bids. This could happen, for instance, if the contracting agent decides to use an advertised resource that is permanently available in the market. For each task in the call-for-bids the bid manager must specify (i) an early start time, (ii) a late finish time, and (iii) the set of tasks that must precede it, and a decommitment penalty. This is the penalty the supplier has to pay to the contractor agent if the supplier decommits.

2. Receive and Evaluate Bids. When bids are returned, the bid manager must assemble them into a minimum-cost feasible schedule in order to determine which bids to accept.

For each bid, the bid manager has the option of selecting the entire bid and paying an overall discounted price, or selecting a subset of the task bids from the bids. Timing information for a bid includes early start, late finish, and duration for each task in the bid. Bids that cover multiple tasks are required to specify prices for each of the individual tasks, as well as a (possibly discounted) price for the entire set of tasks. The semantics of a bid is that a supplier is willing to perform the task or combination of tasks for the bid price, starting at any time in the time window specified in the bid.

Because of the wide expected variability in problem size, and because of the need to be able to limit the time spent evaluating bids, the evaluator uses an adaptive strategy. If the problem complexity is low, a systematic search is used. Bids that are incompatible with each other, either due to overlapping coverage or temporal infeasibility, are marked. If task coverage remains possible after this step, then combinations that meet task coverage and temporal feasibility constraints are evaluated for cost and risk. If the combination of plan size, number of bids, and number of tasks per bid is beyond the systematic-search threshold, then a heuristically-guided simulated-annealing search [12] is used to generate bid combinations for evaluation. More details are reported in [3].

3. Accept Bids. After building the schedule, the bid manager sends bid acceptance messages to the vendors of accepted bids, specifying which parts of which bids are accepted. This completes the negotiation process.

5.3 Contract supervisor

The contract supervisor is responsible for overseeing execution of the plan as contracted, and making decisions on how to respond when events do not proceed as expected. It receives the task assignments from the bid managerm and, through the market, receives updates on plan execution from contracted vendors. It maintains the time map with tasks, vendor commitments, and temporal constraints among tasks. For each event, it must decide whether to respond, and if so, whether to respond directly to a particular vendor, whether to re-bid a portion of the plan, or whether to re-plan and re-bid one or more subgoals of the plan.

The bid manager produces a set of task assignments, each of which includes one or more tasks from the plan, along with the contract data for execution of that task. Contract data includes the task, the resources committed to carrying out that task, an agreed-upon price, an agreed-upon time window and temporal constraints, and the supplier and contractor decommitment penalties.

All the activities of the contract supervisor revolve around the maintenance of the time map. The time map can be thought of visually as a Gantt chart, decorated with contract data and temporal constraints among tasks. For each task the time map records an early start, a late finish time, an expected duration, and the set of precedence constraints.

The early start time for a task is computed by the Critical Path algorithm. In other words, the derived early start time for a task s is the later of the early start time of s computed by the Critical Path algorithm and the beginning of the time window in the accepted bid for the s. Similarly, the late finish time for a task s is the earlier of the late finish time for task s computed by the Critical Path algorithm and the end of the time window in the accepted bid for s. The *slack* for task s is then computed.

6 Interactions of the Contractor Agent with the Market Session

As time passes and the execution of the plan proceeds, the contract supervisor works in conjunction with the market session to drive the plan to completion. In general, the session is responsible for releasing tasks to the suppliers when their prerequisites are satisfied, and for assessing decommitment penalties when the parties fail to satisfy their commitments. In the process, the session forwards to the contract supervisor notifications of task release and task completion events. The contract supervisor is then responsible for making decisions and taking appropriate action in response to those notifications.

In order to carry out its role, the market session maintains a *performance monitoring table*, which is essentially a stripped-down version of the time map maintained by the contractor agent. It contains, for each contracted task, the winning bidder, the early start and late finish times, the contracted subset of successor tasks, and the decommitment penalty function. The performance monitoring table is maintained by updating the expected release time for each successor task whenever the expected finish time of a task changes. When the slack value becomes critical ($slack(s) = 0$), or infeasible ($slack(s) < 0$) the contractor is notified to enable it to take appropriate action.

Each supplier agent is also notified by the market session of task releases, and of changes to expected task release times for each task for which a bid has been awarded.

Following are the classes of events to which the contract supervisor must respond, and a brief outline of the response options.

Nominal Completion. No action is required.

Early Completion. If a critical path is affected, and if the value of the plan could be improved by changing the earliest start time for some task s, then the contract supervisor could request the vendor of s whether the schedule can be moved up, and for what cost. After evaluating the cost/benefit tradeoff, the contract supervisor will request schedule changes accordingly.

Vendor Decommitment. When a supplier decommits, the contract supervisor has three choices: (1) contractor decommitment, (2) attempt to re-bid decommitted task(s), (3) attempt to re-plan and re-bid unsatisfied subgoal(s). The choice that is expected to maximize the profit is the one that will be made.

Missing Event. Completion events are considered missing if their failure to arrive triggers violation of a temporal constraint. This is considered non-performance on the part of the vendor. The contract supervisor responds by notifying the market session of vendor non-performance.

Late Completion. A late completion event is one that occurs later than promised but does not violate temporal constraints. It is a configuration option whether to treat this as a missing event. If not, the contract supervisor responds by re-evaluating the critical path and notifying vendors of affected tasks of changed time windows. All such time window changes will result in tighter windows.

Notice of Late Completion. If a vendor wishes to extend a deadline, it must initiate a negotiation with the contractor using a Notice of Late Completion, giving a new expected completion time and an updated bid. The contract supervisor must then choose whether to accept the updated bid, with the new time commitment, or whether to treat this event as a contractor decommitment.

7 Related Work

Markets play an essential role in the economy, by facilitating the exchange of information, goods, and services [1], and there is a growing need for agents capable of more sophisticated automated negotiations [2, 7]. Automated contracting protocols generally assume direct agent-to-agent negotiation [14]. In our work, agents interact with each other through a market.

To the extent that we require the existence of an external market mechanism as an intermediary, our proposed framework is similar to that of Wellman's market-oriented programming used in AuctionBot [19]. Our framework provides explicit market mechanisms which support more complex interactions. Furthermore, these market mechanisms enforce general market rules and "social laws", such as government regulations, by which all participants must abide.

Sandholm's agents [14, 15] redistribute work among themselves by a contracting mechanism. Sandholm considers agreements involving explicit payments, but he also assumes that the agents are homogeneous – they have equivalent capabilities, and any agent can handle any task. We do not make any of these assumptions. Our agents are heterogeneous, and decide on their own what tasks to handle by responding to a call for bids that requires specific tasks or products within a specified time window.

Various classes of scheduling problems have been considered in the Operations Research literature [9, 10, 6]. Many interesting scheduling problems are computationally intractable, and numerous heuristic approaches have been described [8, 13]. Much of the recent work in scheduling has focused on the problem of maintaining or updating an existing schedule in the face of changes [20, 16]. All of these systems are concerned with determining schedules for individual resources; the assumption is that there is some set of tasks to be done, and some set of resources available to do those tasks, and the problem is to find an optimal or near-optimal assignment of tasks to resources over time. The contractor agent we describe is not resource-limited; these scheduling approaches will be needed for the supplier agents, but they are not well suited for the problem that the contractor agent must solve.

The problem faced by the contractor agent in our design is that of monitoring a plan and its schedule, and finding ways to repair it when it is broken. Muscettola [11] and Tate et al [18] advocate combining the planning and scheduling problems to deal with this issue. Muscettola's approach is based on maintaining state information over time for a relatively fixed set of resources, and so is not directly applicable to our work.

8 Status of the MAGNET Project

The MAGNET market infrastructure has been implemented as a distributed server written in Java, and significant portions of the Bid Manager portion of a Contractor agent have been implemented and tested [3]. We expect to adapt an existing non-linear planner to handle the planning tasks, and we are studying several options for the market Ontology [17]. The resource management portion of an agent in the Supplier role could be met by existing scheduling methods [16, 18, 11]. Our current research activites are concentrated on the negotiation mechanisms and decision procedures in the Bid Manager and Execution Manager portions of the Contractor role.

References

[1] Yannis Bakos. The emerging role of electronic marketplaces on the Internet. *Comm. of the ACM*, 41(8):33–42, August 1998.

[2] Carrie Beam and Arie Segev. Automated negotiations: A survey of the state of the art. Technical Report CITM Working Paper 96-WP-1022, Walter A. Haas School of Business, 1997.

[3] John Collins, Makesim Tsvetovat, Rashmi Sundareswara, Joshua Van Tonder, Maria Gini, and Bamshad Mobasher. Evaluating risk: Flexibility and feasibility in multi-agent contracting. In *Proc. of the Third Int'l Conf. on Autonomous Agents*, May 1999.

[4] John Collins, Ben Youngdahl, Scott Jamison, Bamshad Mobasher, and Maria Gini. A market architecture for multi-agent contracting. In *Proc. of the Second Int'l Conf. on Autonomous Agents*, pages 285–292, May 1998.

[5] Thomas L. Dean and Drew V. McDermott. Temporal data base management. *Artificial Intelligence*, 32:1–55, 1987.

[6] M. R. Garey, R. L. Graham, and D. S. Johnson. Performance guarantees for scheduling algorithms. *Operations Research*, 26(1):3–21, January 1978.

[7] Robert H. Guttman, Alexandros G. Moukas, and Pattie Maes. Agent-mediated electronic commerce: a survey. *Knowledge Engineering Review*, June 1998.

[8] Othar Hansson and Andrew Mayer. Dts: A decision-theoretic scheduler. In Monte Zweben and Mark S. Fox, editors, *Intelligent Scheduling*, pages 371–388. Morgan Kaufmann Publishers, San Francisco, CA, 1994.

[9] Frederick S. Hillier and Gerald J. Lieberman. *Introduction to Operations Research*. McGraw-Hill, 1990.

[10] J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of scheduling under precedence constraints. *Operations Research*, 26(1):22–35, January 1978.

[11] Nicola Muscettola. HSTS: Integrating planning and scheduling. In Monte Zweben and Mark S. Fox, editors, *Intelligent Scheduling*, chapter 6, pages 169–212. Morgan Kaufmann, San Francisco, CA, 1994.

[12] Colin R. Reeves. *Modern Heuristic Techniques for Combinatorial Problems*. John Wiley & Sons, New York, NY, 1993.

[13] Y. Ronen, D. Mosse', and M. E. Pollack. Value-density algorithms for the deliberation scheduling problem. *SIGART Bulletin*, 7(2), 1996.

[14] Tuomas Sandholm and Victor Lesser. Issues in automated negotiation and electronic commerce: Extending the contract net framework. In *1st Int'l Conf. on Multiagent Systems*, pages 328–335, San Francisco, 1995.

[15] Tuomas W. Sandholm and Victor Lesser. On automated contracting in multi-enterprise manufacturing. In *Distributed Enterprise: Advanced Systems and Tools*, pages 33–42, Edinburgh, Scotland, 1995.

[16] Stephen F. Smith. OPIS: A methodology and architecture for reactive scheduling. In Monte Zweben and Mark S. Fox, editors, *Intelligent Scheduling*, chapter 8, pages 29–66. Morgan Kaufmann, San Francisco, CA, 1994.

[17] Austin Tate. Towards a plan ontology. *Journal of the Italian Artificial Intelligence Association*, January 1996.

[18] Austin Tate, Brian Drabble, and Richard Kirby. O-plan2: An open architecture for command, planning, and control. In Monte Zweben and Mark S. Fox, editors, *Intelligent Scheduling*, chapter 7, pages 213–240. Morgan Kaufmann, San Francisco, CA, 1994.

[19] P.R. Wurman, M.P. Wellman, and W.E. Walsh. The Michigan Internet AuctionBot: A configurable auction server for human and software agents. In *Second Int'l Conf. on Autonomous Agents*, May 1998.

[20] Monte Zweben, Brian Daun, Eugene Davis, and Michael Deale. Scheduling and rescheduling with iterative repair. In Monte Zweben and Mark S. Fox, editors, *Intelligent Scheduling*, chapter 8, pages 241–256. Morgan Kaufmann, San Francisco, CA, 1994.