An experimental design to study decommitment in a collaborative multi-agent system in a scheduling domain

Michele Van Dyne[1,*], *Costas* Tsatsoulis[2]

[1]Computer Science Department, Montana Tech, 1300 W. Park St. Butte, MT 59701
[2]Department of Computer Science and Engineering, University of North Texas, 1155 Union Circle #310440, Denton, TX 76203-5017

Abstract. Based on previous multi-agent research, this paper describes the experimental design of extending that research to an additional domain. The original work used collaborative agents, each controlling a radar, to negotiate with each other to perform the task of target tracking. Agents were allowed to schedule their own tasks, commit to tasks requested by other agents, and then, dependent on experimental condition, either always honor commitments, unilaterally decommit if another higher priority commitment occurs, or negotiate whether to drop a commitment or not is another apparently higher priority request is received. In the original research, the domain was highly time constrained, and agents were required to operate in a real-time environment. This research extends the concept of agent decommitment into a resource scheduling domain. The problem is no longer time constrained, however, the number of agents is increased dramatically. In this paper, we describe the theoretical background and experimental design of the new research.

1 Background

Previous multi-agent research has assumed that in collaborative multi-agent systems when an agent makes a commitment, then it would honor it. However, there may be situations where circumstances change and honoring the commitment would lead to a lowering of the social welfare of the system; in such situation it is preferable not to honor a commitment. In the multi-agent systems of interest agents are multitasking and can perform certain actions in parallel. Since the agents are collaborative they must be able to request that another agent perform a task, and thus are capable of communication. When an agent in the system attempts to achieve a goal, it decides to perform a set of tasks to reach the goal. Each decision to perform a task is a commitment to the task. Our work follows the Belief-Desire-Intention (BDI) model of agent architecture defined by Rao and Georgeff [1], where the commitment to a task is viewed as an "intention," and the goal that must be attained is viewed as a "desire" in the BDI framework. Finally, the information available to the agent is known as "beliefs" in [1][2] and indicate to the agent the feasibility of the task as well as the subsequent attainment of the goal. An agent may also commit to a task because another agent has requested it. Thus, we have task commitments either to oneself or to another agent.

Schut and Wooldridge extend the BDI model to include a decision theoretic approach to what they call "intention reconsideration" [3]. Their approach allows an agent to decide for itself whether to perform an action or to deliberate over its intentions. Two conditions are defined under which reconsideration may be appropriate: when conditions have changed such that an intention can no longer be achieved, and when better opportunities arise. While this approach moves the decommitment decisions from design-time to runtime, it only allows an agent to consider what should be done "next", a default action or a reconsideration. This approach studied only the effects on a single-agent system, and thus disregards effects on a multi-agent system.

Decommitment may happen if a task becomes impossible to complete, or if an alternate task that has is of higher importance is presented, or, finally, if conditions indicate that the performance of a task is no longer a productive or needed activity. The goal of our work was to study what would be the best way for an agent to drop a previous commitment to perform a task for another agent in the multi-agent system. We wanted to study the effects of two different types of decommitments to the overall social welfare of the multi-agent system: (1) unilateral, where the agent drops a task to which it has committed itself without informing the other agent, and (2) negotiated, where the agent will inform the other agent that it is planning to drop its commitment, but it is open to negotiate and be convinced not to do so. Finally, we compared the two decommitment approaches to the effects on the social welfare of the system if no commitments were ever dropped.

To perform our previous research, we developed an architecture of a multi-agent system that operated in an environment of hard real time, using sensors that had to

* Corresponding author: mvandyne@mtech.edu

collaborate to track multiple targets [4]. An agent controlled a sensor and had to commit to tracking a target over a period of time, and then switch to another target, and so on. If a more important target, though, appeared, it may have to decommit from tracking a previous target and move to the new one. Commitments were made to the first agent to sense a target, as multiple agents needed to track in order to establish a precise target location. Since targets would move out of a sensor's area, commitments had to be flexible and fluid, and move with the target across the sensor and agent network.

We found the architecture to be very useful in creating multiple decommitment experiments by varying the number of targets and sensors in the environment, as well as the target speed [5]. We computed the social welfare of the multi-agent system as the utility of the target tracking schedule of the whole network, which is a function of accuracy, number of measurements per target per second, and the balance of managing to track multiple targets.

Using our experimental set up we demonstrated that both unilateral and negotiated decommitment performed as hypothesized, that is, overall performance improved, and that negotiated decommitment provided additional benefit over the performance of unilateral commitment. Graceful degradation of performance was also demonstrated in all conditions as the constraints on the system. Finally, under all conditions, agents demonstrated rationality in decision making and both negotiated decommitment and unilateral decommitment provided a higher locally assessed utility than did the baseline condition.

Our previous research provided an architecture on which to test the concept in different domains. In particular, the fact that agents may hold resources that other agents need opens to possibility of using this architecture in the domain of scheduling. In work by Sen and Durfee, an agent schedules tasks to be performed at a given point in time. These tasks may originate from the agent itself, or may be committed to at the request of another agent. The distributed task scheduling problem is defined in [6] as a group of tasks to be scheduled requiring the use of a set of resources controlled by a set of agents. The design described in this paper uses the domain of classroom scheduling for a university. Thus, the requirement that a task be scheduled at a specific time can be dropped, however the framework of agents holding resources that are needed by others remains a criterion.

2 The Problem Domain

Classroom scheduling at the university being used as a test case uses a process where course offerings scheduled in a particular room at a particular time for the corresponding previous semester have priority in continuing to use that room in subsequent semesters. If no course offerings changed, this approach would yield a conflict-free schedule every semester. Unfortunately, course offering requirements do change, making it necessary to change at least a portion of the previous schedule.

The possible changes to a course offering that will affect future schedules are:

- Offering has been cancelled
- New course offering added
- Additional sections of a course added
- Expected enrollment in the course has changed (either increased or decreased)
- Time/days course is offered has changed
- Classroom requirements (equipment/facilities) have changed

The following sections discuss the effect on the scheduling process when each of the conditions has occurred.

2.1 Course Offering Cancelled

A course that was offered in a previous semester schedule that has been cancelled in the future schedule frees up a resource (classroom/days/time) that can be used for other courses. In the architecture of the system, this creates a free resource as opposed to one that is currently held by a current course offering agent. Free resources will be consumed before a course scheduling agent requests that another one give up its holdings, if at all possible.

2.2 New Course Offering Added

When a new course is added to the schedule, it holds no resources from a previous semester. Generally, a newly added course will have a request for a specific set of days and a time, along with an estimate of the enrollment, and a request for a particular room, along with particular requirements for the offering, including capabilities and facilities present in the room. If that room is a free resource at that time, the request can be fulfilled as-is. If not, however, negotiation between the agent holding that resource from the previous schedule, and the agent requesting the resource as a newly added course will be necessary.

2.3 Additional Sections of a Course Added

Adding additional sections of a course is very similar to adding a new course. These new sections hold no resources from a previous semester and generally will request a specific set of days and a time, along with an estimate of the enrolment, and a request for a particular room, the same as with a wholly new course offering. As before, if that room is a free resource at that time, the request can be fulfilled as-is. If not, however, negotiation between the agent holding that resource from the previous schedule, and the agent requesting the resource as a newly added course will be necessary.

2.4 Expected Enrollment Changed

If the expected enrollment of a course decreases, it may be possible to move that course offering to a smaller room, if one is available. The previous course offering still holds the resources it was schedule for, but now has less priority over those resources.

On the other hand, if the expected enrollment increases and the increase is too much for its previous room to handle, one of two things can happen. The course enrollment may simply be capped by the room capacity, or the course offering experiencing an expected increase can negotiate with other course offering agents for a larger room.

2.5 Time/Days of Offering Changed

On occasion, the department offering a course may need to change the days/time that a course is offered so that it fits the schedule of the students taking the course, or so that it fits the schedule of the instructor teaching the course. Individual academic departments keep track of the other courses students in a cohort are normally taking and thus has an idea of when offerings can be scheduled to work for that cohort. Additionally, individual departments are aware of the teaching schedules of their instructional faculty, and can work out when a course offering can be scheduled to meet those requirements also.

Individual academic departments look at day/time changes from a local viewpoint, however. They do not have an overview of what rooms may be available at given times, nor the requirements of courses offered by other departments.

2.6 Classroom Requirements Changed

Changes in how a course is taught can change the classroom requirements of that course. This may include requiring computing workstations for each student, or other laboratory equipment. It may also include changes to the instructor's podium equipment, including computing and projection. Some of these changes may be possible to accommodate by enhancing the equipment of the currently scheduled classroom, while others may necessitate the change to a different room already so equipped.

3 Architecture

Our previous work was concerned with individual independent agents each holding access to a resource (radar) which may be used to assist other agents. To map this architecture to the current domain, we can view each course offering in a previous schedule as an agent holding a resource, and each changed course as an agent requesting resources from those agents holding them. In our previous work, agents were constrained by timing, and had to maintain a task schedule during which time a request had to be honoured (that is, a radar sector must be enabled at a specific time in order to track an expected target). In the current domain, this timing constraint is removed, so individual agents do not need to maintain a task list. Nor do agents in the current domain need to respond to requests in a hard real time limit as they did in the previous work.

The abstracted goal of the overall agent system is, however, the same as before. Agents in the new scheduling domain collaborate to maximize the overall social welfare of the system, just as they did in the previous domain. One major difference, however, is the number of agents. In the previous work, we experimented with 4-12 independent agents controlling radars. In the current work, with each course offering and changed course acting as an agent, we have on the order of 1500 agents. Fortunately, only a small subset of these will be involved in negotiations – only those that have a conflict between a currently held resource and a requested one.

3.1 System Architecture

The data for this problem resides in three data sets. The first is the complete schedule for all course offerings held in the previous semester corresponding to the current one. The second is the list of requests for all (or at least most) of the course offerings to be held in the upcoming semester. In both of these datasets, the requirements of the course offering for the resource are contained in the data. The final dataset contains information about all rooms and their capacities and facilities.

The first pass of the overall system is to identify all course offerings that are exactly the same in the previous and current schedule and eliminate those from the "request" dataset since they pose no conflicts. These are stored in a new dataset – the "potential" schedule. The next pass is to look at requests that are different from the previous schedule. Any of these that do not cause a conflict may be added to the potential schedule. What remains to be dealt with are the requests that do cause conflicts.

An agent is spawned for each of the requests that cause conflicts along with an agent for each of the courses on the potential schedule with which they conflict. Note that a single course offering request may conflict with more than one offering on the potential schedule.

Following the formalization introduced by Soh and Tsatsoulis, the social agent environment can be expressed as follows [7]:

Ω a multi-agent system

Ψ a "neighborhood" in the system

$\lambda(\alpha,\beta)$, predicate indicating agent α knows about agent β.

A neighborhood is defined as:

$$\Psi \subseteq \Omega \ , \ \Psi \neq \varnothing \qquad (1), (2)$$

Every agent, α_i in neighborhood Ψ knows about all other agents in that neighborhood. That is:

$$\lambda(\alpha_i, \alpha_j), \forall i \forall j \alpha_i, \alpha_j \in \Psi \qquad (3)$$

There may be any number of neighborhoods, they need not have the same number of members, and neighborhoods may overlap. Thus,

$$\Omega = \{\Psi_1, \Psi_2, \ldots, \Psi_N\} \qquad (4)$$

Agents may communicate with other agents within their neighborhood, that is, an agent may only communicate with those that it is aware of. Thus negotiation is also restricted to agents within the same neighborhood.

Neighborhoods are defined not necessarily by proximity, but by areas of conflict. For example, all agents negotiating for the use of rooms with the same capabilities at the same times can define a neighborhood, as can agents negotiating for a specific space. Neighborhoods are not static, and as commitments to relinquish or exchange resources are made, neighborhood boundaries may change. The implication of this constraint is that broadcast requests are not made to all agents, reducing the communication load on the system as a whole. The overhead of recalculating neighborhoods is considered inconsequential in comparison to the cost of mass communications.

3.2 Agent Architecture

Each agent has the capacity to communicate with other agents in its neighborhood, to negotiate the relinquishment or exchange of resources with another agent, and to make final decisions on whether to grant requests or not. Each of these processes is carried on by individual threads within the agent, while each agent is implemented as an independent process.

Similar to our previous work, a priority is assigned to the resources held by a particular agent and to the agent requesting that resource. The difference in this research is that it is the importance of holding a resource to a particular resource that is prioritized, and not the priority of performing a particular task.

A resource priority, with respect to a given agent, is represented by the tuple:

$w_i = (p_i, vh_i, c_i, wh_i)$, where:

p_i is the calculated priority of a particular resource (defined below);

vh_i is the agent $A_i's$ assessment of the validity of the requesting agent $h_i's$ information;

c_i is the constrainedness of the task, comprised of the number of other agents also in the conflict set and the specificity of resource requirements for that offering;

wh_i is $h_i's$ assessment of the priority of its need for the resources;

The calculated priority, p_i, of a particular resource is a combination of the capabilities and facilities of a particular classroom and the needs of the current agent. The closer the resource to the needs of the agent, the higher the priority of keeping that resource, and the further the resource to the needs of the agent, the lower the priority. For example, if an agent needs a room with a capacity to hold 50 students and requires a podium computer and projection system, and the room it currently holds has the capacity for 150 students and has PCs at each student station and has a podium computer and projection, the resource clearly meets the need of the agent, but also exceeds the minimum need, therefore, it will have a lower calculated priority to that agent.

Agent negotiation is initiated by a requesting agent to an agent holding resources it desires. The initial request contains the requesting agent's evaluation of its own priority in having the resources held by the agent currently in possession of those resources. If the possessing agent agrees that the priority of the requesting agent is higher than its own priority, the "negotiation" between these two agents can stop at this point and the possessing agent tentatively relinquishes its resources and becomes a requesting agent. If this new requesting agent is successful at finding resources to meet its needs, then it fully relinquishes its former resource and the conflict is resolved. If not, however, agents must do additional work.

Clearly, the requesting agent is the one who must do the bulk of the work. If another agent is unwilling to give up resources currently held, the requesting agent must then investigate other options. These can include finding another room that will meet its needs, changing the day/time of the course offering, or possibly splitting the offering into smaller sections (essentially spawning two or more processes with less restrictive needs). If any of these activities is successful, the conflict is resolved and no additional negotiation is needed. If not, the requesting agent can go back to the possessing agent again, now with a higher priority to its request since it has investigated other avenues and has found no possible solution. If this requesting agent is one who had formerly held resources and has only tentatively relinquished them to another requesting agent, it may choose to decommit those resources and take them back for itself, or it may choose to continue negotiations with other agents, dependent on the conditions of the experiment begin run.

4 Experimental Design

The ultimate goal of the agent system is to find a schedule of course offerings that meets the needs of all,

that is, there are no conflicts, and all course offerings are holding adequate resources. This defines the optimal measurement of the overall social welfare for the system. Therefore, the actual overall measurement of social welfare is a function of the best possible scenario less factors for any unscheduled course offerings and any course offerings assigned resources that are less than optimal. Individual agents, or even groups of agents in a conflict set, are unable to calculate this overall social welfare score, but do so with local approximations in attempting to negotiate with each other to secure resources

There are three conditions to be tested to determine the potential of commitment/decommitment in producing optimized solutions. The first condition is that once an agent relinquishes resources, it does so completely rather than tentatively, and thus becomes a true requesting agent. This condition is the baseline in which a committing agent always honors its commitments. The second condition is where an agent makes a tentative commitment, investigates whether it can secure resources elsewhere, and if not, simply drops the tentative commitment. This is the unilateral decommitment condition in which the requesting agent has no recourse but to accept the unilateral decision of the possessing agent. The final condition is where the possessing agent, having made a tentative commitment to a requesting agent, but not being able to secure other resources, goes back into negotiation with the original requesting agent rather than simply dropping the commitment. This is the negotiated decommitment condition.

Datasets representing the three data sources (prior schedule – fall semester 2016, current requests – fall semester 2017, and room capabilities) have been secured for testing purposes. The experiment will be run on a High Performance Computing (HPC) platform which has 22 nodes each with 32 process elements per node, for a total of 704 process elements. Two of the nodes have Nvidia Tesla K20 GPUs installed, and with these, we can provide an additional 2496 cuda cores per K20.

5 Future Work

While this paper has described the theoretical basis for applying a multi-agent system with decommitment parameters, and has outlined the experimental design to be used to test the system, the actual experimentation is yet to be performed. Results of the experimentation will be reported in a future paper.

In addition to the work outlined in this paper, extensions to the system include factoring in additional scheduling constraints. Course instructors may have preferences and requirements that are unrelated to the course offering itself. Preferences might include classrooms that are near that instructor's office, not teaching courses back to back, and if so, back to back courses should be near each other, and familiarity with a particular room and its operation. Requirements may include handicap accessibility among other things.

Acknowledgements

The authors would like to thank Montana Tech Enrollment Services staff Leslie Dickerson, Registrar; Janet Friesz, Supervisor; and Abigail Peterson, Administrative Evaluator III, for their assistance in obtaining course offering and scheduling test data for use in this research.

References

1. A. S. Rao and M. P. Georgeff. 1995. "BDI Agents: from theory to practice", Proceedings of the 1st International Conference on Multi-Agent Systems (ICMAS '95), June 12-14, San Francisco, CA, pp. 312-319.
2. Parsons, T.S., Sierra, C. and Jennings, N.R., Agents that reason and negotiate by arguing, Journal of Logic and Computation, Vol. 8 No. 3, 1998, pp. 261-292.
3. Schut, M. and Wooldridge, M., Principles on intention reconsideration, Proceedings of the Fifth International Conference on Autonomous Agents, 2001, pp. 340-347.
4. M. Van Dyne and C. Tsatsoulis. 2010. "A Comparison of Agent Decommitment Techniques in a Real-Time Environment", Proceedings of the 9th Conference on Artificial Intelligence, Knowledge Engineering, and Databases (AIKED 2010), University of Cambridge, UK, Feb. 20-22, pp. 271-279.
5. M. Van Dyne and C. Tsatsoulis. 2011. "Effect of agent decommitment in a target tracking domain", Proceedings of the 2011 IEEE Aerospace Conference, Big Sky, MT, Mar. 5012.
6. Sen, S. and Durfee, E., A contracting model for flexible distributed scheduling, Annals of Operations Research, Vol. 65, 1996, pp. 195-222.
7. L-K. Soh and C. Tsatsoulis, A Real-Time Negotiation Model and a Multi-Agent Sensor Network Implementation, Autonomous Agents and Multi-Agent Systems, Vol. 11, No. 3, 2005, pp. 215-271.