Market Based Adaptive Resource Allocation for Distributed Rescue Teams

Guruprasad Airy
The Pennsylvania State University
gairy@ist.psu.edu

Tracy Mullen
The Pennsylvania State University
tmullen@ist.psu.edu

John Yen
The Pennsylvania State University
jyen@ist.psu.edu

ABSTRACT

The dynamic nature of real-world rescue scenarios (e.g., military, emergency first response, hurricane relief) requires constant reevaluation of resource assignments. New events can trigger additional resource requirements generating conflicts about how to reassign resources across tasks in an emerging crisis. Reallocation is further complicated as some resources are synergistic (i.e., helicopter and pilot) and many distributed rescue teams have limited information about other teams' status. We show how integrating a team-based multi-agent planning system with standard combinatorial auction methods to dynamically re-allocate resources can maximize overall rescue utility while providing for graceful managed degradation under conditions of extreme stress. The key innovation of our approach is that we explicitly provide a framework that incorporates the costs involved in dynamically switching resources from one task to another. We compare our system's performance against two other approaches.

Keywords

Resource Allocation, Combinatorial Auctions, Multi Agents, Planning.

INTRODUCTION

Multi-agent systems have been applied extensively to the rescue domain, most of these focus on problem solving such as flood rescue (Cuena, et al.,1999), escape from a theater (Mita et al., 2000) and Robocup Rescue (Kitano, et al., 1999). However, they do not focus on the problem of resource allocation in these dynamic uncertain situations to the rescue teams. Our focus is on distributed teams from different organizations engaging in a coordinated response to rescue situations who share a common resource pool. For dynamic crises, these common resources may need to be re-allocated when new events occur to accommodate additional teams with new tasks. In addition, since resources may interact in synergist ways, they often need to be considered as resource bundles rather than individually. For example, a helicopter and a pilot are synergistic or complimentary resources in the sense that one without the other is much less useful (if not useless).

The value, or utility, of any task to the overall mission is conditional on how it is achieved. If speed is of the essence, then the fastest method may have a higher value than others. Ideally, teams should be able to express preferences for resources that 1) encompass tradeoffs between alternative means of achieving their tasks (e.g., use a truck vs. use a helicopter) and 2) can express resource bundles (e.g., truck and a driver). For example, suppose the task of delivering food can be achieved either with a method that requires a truck and driver or one that requires a helicopter and pilot. If it is more important that food get delivered cheaply rather than quickly, then the truck and driver delivery method utility will be higher than the helicopter and pilot method. However, to evaluate the overall utility, local decision makers must also consider the *switching costs* involved in reallocating resources from one team to another. For example, if a team needs a truck, but the truck is far away from the team's current location, then it may not be worth the time and energy required to move the truck. Instead, the team may be better off using the alternative method (e.g., a helicopter) to accomplish their task.

Although centralized mechanisms to solve the optimization problem can provide high-quality solutions, they are often not scalable and sending all the necessary information can result in a communications bottleneck. Distributed solutions can be more flexible and more reliable; although those benefits are obtained via extra efforts in coordinating multiple agents. However, often distributed agents map naturally into the problem domain where task information is localized across different teams or even organizations. Market-based approaches offer an inherently distributed mechanism that can compare "apples" and "oranges" using the

common numeraire of money, thus reducing communication overhead to the single dimension of price. Under certain assumptions, price systems have been proven to provide the minimum dimensionality of messages necessary to determine Pareto-optimal allocations (Jordan, 1982).

In this work, we integrate a market mechanism with a planning system to create our multi-agent architecture. Each agent team is assigned a task, where each task specifies one or more alternative methods of task execution and each method specifies the resources required. Each team informs the system about the value it places on each of the alternative methods of accomplishing its task and its current set of resources. The system then computes the cost of switching resources from one team to another and reassigns resources to tasks so that the overall rescue mission's utility is maximized. Using this framework, teams do not have to share all of their planning details, just the resource needs for each task method and each method's utility. This simplifies team coordination across distributed locations and organizational boundaries.

We compare our resource coordination system with two other approaches under different stress conditions as well as against an (over)estimate of the optimal allocation, and show how the global utility and task completion percentage varies under each approach and type of system stress. We show that our approach provides the best overall mission utility, including graceful managed degradation when the system is under high stress. We compare all of these solutions to the myopic optimal solution. This solution assumes that there are no switching costs and optimally allocates resources for each time period. A non-myopic optimal solution is computationally infeasible, but in our future work, we will employ an expected likelihood of success framework to capture expectations of future states to improve our inter-temporal resource allocations.

The rest of the paper is organized as follows. We discuss an example scenario in Section 2 and background knowledge in Section 3. Our approach is elaborated in Section 4. We describe the experimental setup and results in Section 5, with conclusions and future work in Section 6.

EXAMPLE SCENARIO

To motivate our approach, we describe the following simplified emergency management scenario. Suppose that a Category 5 hurricane hits a major metropolitan area, and teams must initially accomplish the following two tasks:

1) Delivering foods to a large group isolated people (Utility: 300, resource: Helicopter)

2) Fix levee before further flooding can occur (Utility: 400, resource: Boat).

For each task, the priority level is factored into the expected utility of a given method for achieving the task. The goal of the system at all times is to maximize the global utility, which is the sum of the utilities of all the currently executing tasks.

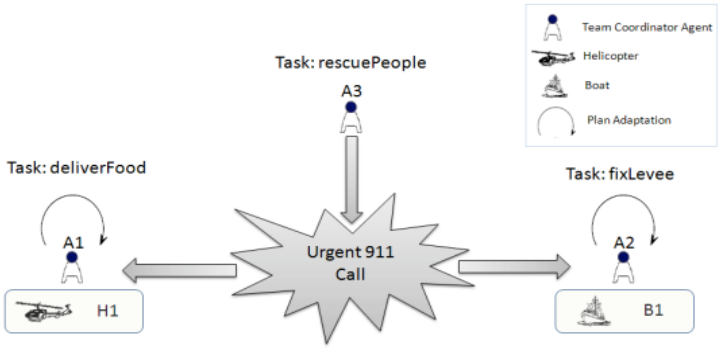


Figure 1. Scenario Illustration

From the example in Figure 1, team coordinator agents A1 and A2 are responsible for achieving tasks deliverFood and fixLevee respectively. Assume that the deliverFood task has been allocated the sole helicopter H1, and the fixLevee task has been allocated the sole boat B1. The total global utility in this case is 700.

There is an emerging task rescuePeople, which requires at least one helicopter or one boat to quickly rescue a group of people. Now there is a dilemma among the agents to decide who should use the limited resources available. When A1 and A2 receive the resource *reallocation* request from the coordinator A3 of the new task rescuePeople, each of them needs to decide whether to switch its own resources from itself to the emerging task in order to maximize the global utility.

Suppose that A3's maximum utility is 500 for one helicopter, and 400 for one boat, its final bid will be (rescuePeople: Helicopter,500) XOR (rescuePeople: Boat, 400), where XOR stands for exclusive-or (i.e., one or the other, but not both). In this example, the outcome will be that helicopter H1 will be switched from the task deliverFood to the task rescuePeople (which has a higher priority/value) and the expected global utility increases from 700 to 900. Note that this reallocation does not consider switching costs.

While extremely simplified, this example illustrates the need for a system that can dynamically make decisions on which tasks should be allocated resources. Markets provide a natural way of expressing the value of different resource requirements to accomplishing a task, where the utility value for resources can be based in part on the priority of the tasks.

BACKGROUND

Our system integrates a team-based planning system with a combinatorial auction mechanism (i.e., an auction that can deal with combinations of resources). We describe the background work in these two areas in the following subsections. While the Robocup Rescue competition (Kitano, et al., 1999), an emergency response competition, shares a similar problem domain if differs in the fact that Robocup Rescue planners use auction methods to allocate tasks to teams based on resource constraints (Nair, et al., 2001; Suarez, et al., 2006). Our work assumes that tasks have already been allocated to teams and now teams must share common resources. The novelty of our approach is that we examine inter-temporal resource reallocation via combinatorial auctions, including the costs involved in dynamically switching resources from one task to another.

Multi-Agent Planning

In multi-agent systems, agents may represent teams each having their own goals or plans, but who find it necessary to address interdependencies between these plans. Thus, multi-agent planning comprises both planning and coordination processes (de Weerdt, et al., 2009). Common coordination processes include coordinated resource allocation for agents sharing common resources, where task assignment can be considered as a special case of coordinated resource allocation (Malone, et al., 1994). However, in our approach, we focus not on assigning tasks, but on selecting one of several alternative methods of achieving a task, each with its own resource requirements. We also focus on dynamic rather than static resource coordination. Dynamic multi-agent planning can be classified into different approaches, including but not limited to,

- continual planning that interleaves planning, execution, and monitoring (Brenner, et al., 2006)
- re-planning that plans again from scratch (Cushing, et al., 2008)
- plan repair that tries to fix the current plan to meet the new situation (van der Krogt, et al., 2005).

In our case, we assume that each agent represents a separate organization whose tasks stay the same. However, new emerging situations and joint resource requirements may arise and their tasks may have to be executed using alternative task methods.

In some sense, our work falls between plan repair and re-planning. Each agent keeps its current task, while the system re-plans how to allocate resources across possible task methods. In doing so, the system implicitly repairs the previous task method selection by selecting the best method to use in the current situation. To avoid thrashing between task methods, we take into account any switching costs necessary to re-allocate resources to new tasks at new locations. Our switching cost measure can be considered analogous to plan stability measures (Fox, et al., 2006) but based on changes in resource costs rather than plan activities.

Market-Based Resource Allocation

Market-based solutions have been studied extensively in the field of resource allocation in recent years, including resource allocation with incomplete information (An, et al., 2007) and severely bounded communication (Blumrosen, et al., 2007). Markets provide a goal-oriented way of allocating resources among competing requesters, while maximizing the overall utility or social welfare of the users (Shneidman, et al., 2005). Markets also provide distributed agents with a relatively low bandwidth mechanism for conveying preference and value information via price as well as provide feedback to each individual planner in a coordinated environment about how others value certain resources at various points in time. This feedback can potentially be used to improve planning process in the future.

While combinatorial auctions can support inefficient equilibriums (Sandholm, 2002) (i.e., the equilibrium need not be an optimal solution), if the agent payoff mechanism makes truth revelation the dominant agent strategy, then the auctioneer can still optimally allocate the goods. Winner determination is an NP-complete problem, but

recent work has resulted in algorithms that are very fast even for large problems (Sandholm, 2002) as well as anytime approximation algorithms (Avasarala, et al., 2006).

TECHNICAL DETAILS

Architecture

For the purpose of this study, we developed a resource sharing architecture based on R-CAST, a collaborative agent architecture based on the concept of Klein's recognition primed decision model (Klein, 1998). We chose this architecture because it enables agents to make decisions in time-critical situations based on prior experiences and execute and co-ordinate a set of distributed tasks based on the decision.

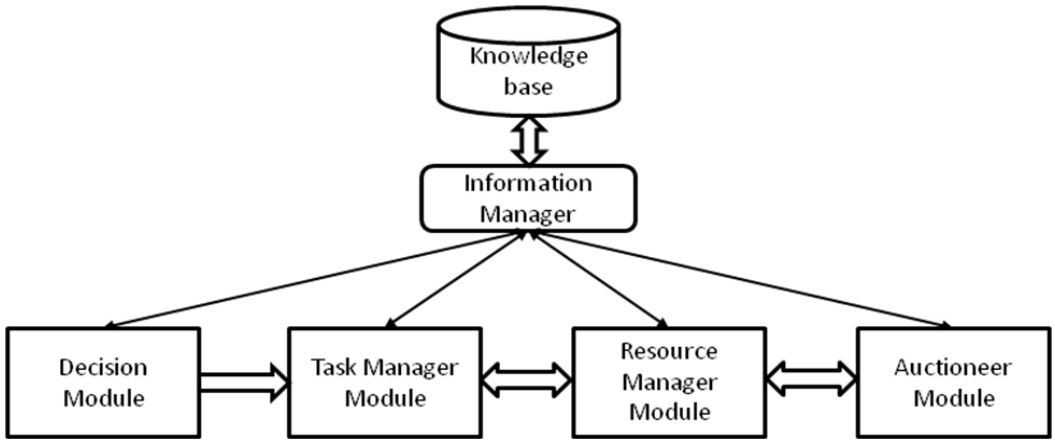


Figure 2. Agent components and their interaction

Figure 2 shows the different components of the agent framework. The knowledge base is a forward chaining rule-based proof-preserving system, which enables the agent to maintain its beliefs regarding the external world and other agents in first order predicate syntax. It communicates with the other modules through an information manager. The decision module analyzes the current situation based on inputs and matches it with the closest experience in its history. The output of the decision module is a task that needs to be executed. The task manager maintains, instantiates, and executes tasks. The task can be triggered either internally (e.g., at a particular time) or externally. The role of the resource manager is to satisfy resource requirements as they arise. The auctioneer component performs the market-based optimization when there are resource requirements.

The decision making and the task execution details of R-CAST are beyond the scope of this paper but are described in (Fan, et al., 2007). While currently integrated with R-CAST, our work can be easily extended to other multi-agent architectures with minor modifications.

Task Representation

Each task is represented in an extended form of the MALLET syntax (Miller, 2006). A task has a set of pre-conditions which need to be met before the agent begins execution of the task. The pre-conditions can be inputs from the outside world or even the statuses of other tasks being executed. In addition, a task has a set of enumerated alternative methods to execute the task as shown in Figure 3. Each task method provides a plan for solving a particular problem. For our experiments, each task method requires at least one resource bundle. The alternative methods for task execution provide flexibility in choosing resources for task execution and allow tasks to switch methods dynamically during resource re-distribution and continue execution.

The utility value represents the value gained if the task is completed using that method. The utility value is generated based on the types and number of resources of each type being used. Assigning utility values is a delicate topic, especially in a rescue scenario which can raise moral and ethical concerns. Preference elicitation methods could be used here to help assess organizational values for a given rescue domain. For the purpose of our research, we assume that the utility values represent the priority of each method. The preference among the alternative methods is a quantitative cardinal preference structure (Chevaleyre, et al., 2006). A complete weak ordering $\leftarrow_u$ can be defined for the set of alternatives, given by $x \leftarrow_u y$ if and only if $u(x) \leq u(y)$, where x and y represent alternatives and $u(x)$ and $u(y)$ represent their utility values respectively.

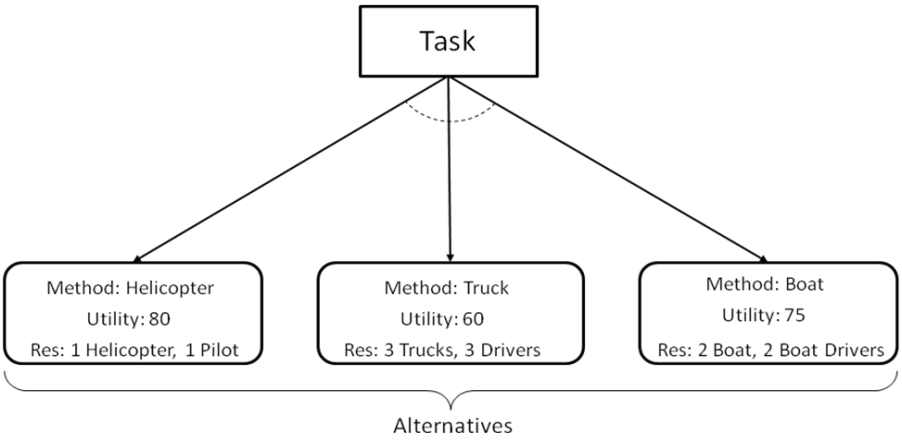


Figure 3. Task Illustration

Resource Representation

Each alternative method has a resource bundle that describes the resources required for that particular method. A resource bundle consists of a conjunctive set of pairs of resource types and the quantity of each resource type required. Each resource bundle can be thought of as a tuple in the form $\{<R_1,C_1>,\ldots<R_n,C_n>\}$ where R_1, R_2, $\ldots R_n$ represent resource types needed and $C_1,C_2 \ldots C_n$ represent the quantity of each resource type respectively. In this research, we assume that resources are indivisible, static, discrete and non-shareable. Also, we assume that this is a multi-unit setting, where multiple resources of the same type are available and the agents cannot distinguish between these resources.

In addition, each agent maintains its belief about the status of all its resources in its knowledge base as a set of predicates. For example,

```
(Resource ?resourceID ?resourceType ?currentTeam ?currentTask)

(ResourceProperty ?resourceID ?propertyName ?propertyValue)

(Resource Heli001 Helicopter TeamA TaskRescue)

(ResourceProperty Heli001 FuelLevel 50.0)

(ResourceProperty Heli001 Speed 100.0)
```

The use of the knowledge base provides flexibility in not only describing various resources and their properties but also allows agents to update their beliefs about the resources dynamically.

Process Flow

Figure 4 shows the process flow in our system. Once the pre-conditions of a task have been satisfied, the *resource needer* agent generates a *resource-request* bid based on the resource requirements of the task and sends it to the *Auctioneer*. The auctioneer forwards this request to all the other agents in the system. On receipt of a resource request bid, the *resource provider* agents check their resource pool and send out *resource-offers* if they have resources that can satisfy the request. The Auctioneer collects all the bids until a specified deadline and determines the optimal allocation of resources. Messages are then sent to the appropriate agents to transfer resources. Task execution begins once the resources have been successfully transferred. Although the auctioneer is centralized for each auction, any agent with an auctioneer module can be dynamically assigned to perform the role of the auctioneer. To lessen the communication overhead, we assign the agent with the resource needs as the auctioneer in this work.

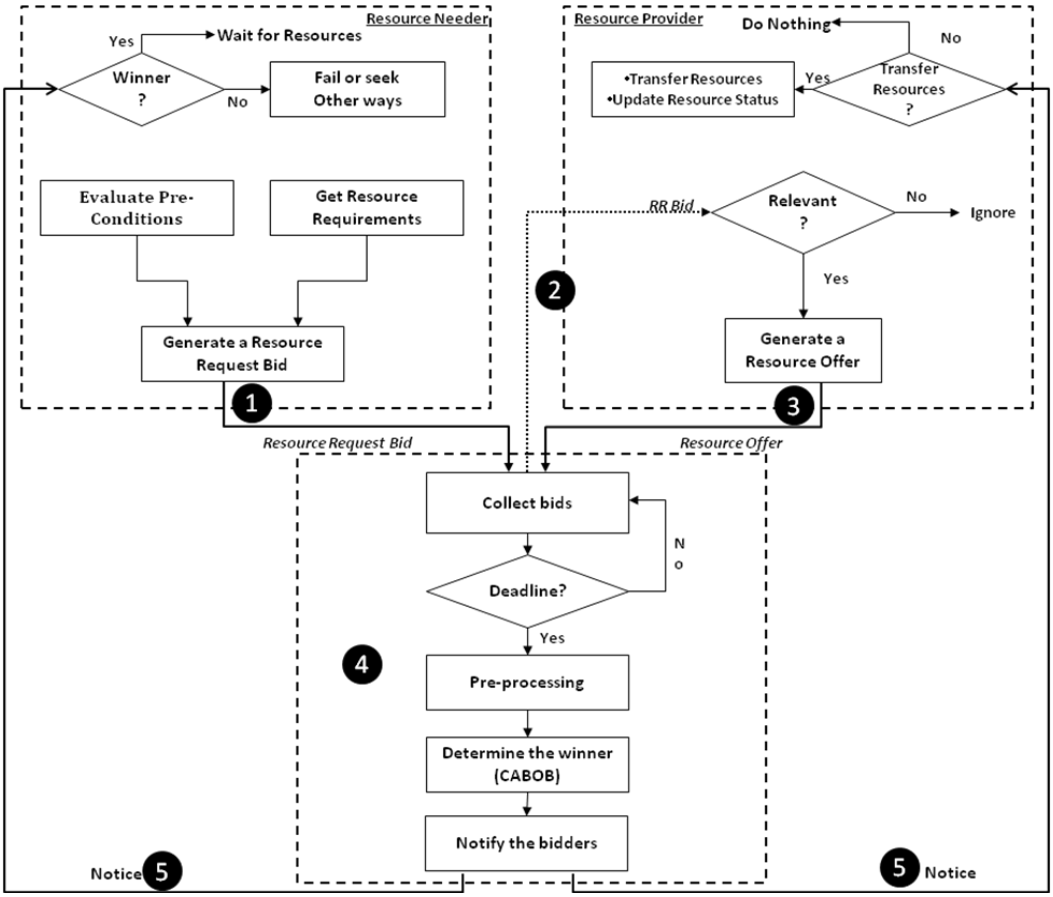


Figure 4. Process Flow

Allocation Procedure

The *auctioneer-agent* elicits bids from all the agents in the system and decides on a final allocation of resources. While there are distributed combinatorial auctions (Mendoza, et al., 2007), for our initial research, we use a centralized combinatorial approach as it simplifies our communication protocols and allows us to use a fast and optimal winner determination algorithm. While the auction is centralized, the system itself is still distributed in the sense that participants simply reveal to the auction how much they value a particular resource and do not have to share any of the "raw" data involved in arriving at that valuation. The following subsections describe the steps in the bidding and auction phase. Figure 5 illustrates the allocation procedure and shows how a resource request bundle combines with a resource offer to form two XOR resource bundle-resource offer combinations and one among them is selected as the winner.

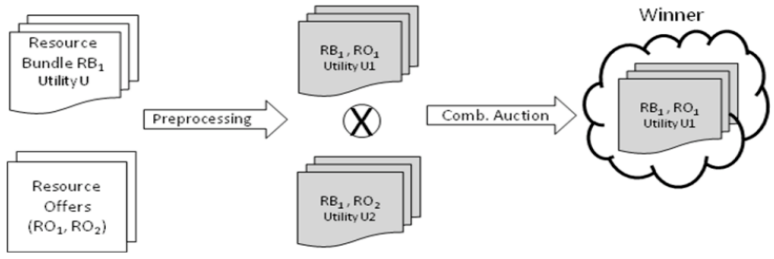


Figure 5. Allocation Procedure

Bidding Protocols

The agent requesting for resources sends out a *resource request bid* to the auctioneer. This bid consists of a set of disjunctive resource bundles needed to execute the current task and represented in the XOR bidding language (Boutilier, et al., 2001).

Once the auctioneer receives a *resource request bid*, it is forwarded to the other agents in the system requesting resource offers that can satisfy the request. Note that the resource request is for a type of resource (such as helicopter) and the resource offer will be for an instance of that resource type (for example, Heli001). Any agent in the system that controls a resource that can satisfy the request will send out a *resource offer* to the auctioneer containing the list of resources and their attributes (location, fuel load, etc) that it can offer. For our current research, we assume that agents bid truthfully.

Resource Adaptation

The resource adaption aspect is a unique feature of our system. If an agent decides to give up a resource in use, it will try to obtain other resources that can satisfy its current task. This is done by sending their *resource requests* in addition to their *resource offers*. Note that these new resource requests may trigger additional resource offers. This recursive bidding continues until the system reaches a stable state or the deadline of the task that triggered that initial auction is reached. While typical times to stabilize are very short, future research may investigate timing and communication tradeoffs between different algorithms for triggering resource offers and requests. The agent using a resource continues to use it until notified by the auctioneer that the resource needs to be switched.

Pre-processing and switching cost

Once all the *resource request* and *resource offer* bids have been elicited by the auctioneer, a preprocessing step considers each resource request and offer and determines the *switching cost* of moving each resource from supporting its current tasking to the new task. The system then updates the utility value of each bid to reflect the switching costs.

Winner Determination

For each of n bids with utility U, the WDP is to maximize the utilitarian social welfare $\sum_{i=1}^{n} U_i$ which is the sum of the individual utilities (Chevaleyre, et al., 2006), subject to the condition that each task T has at most one method resource bundle with a satisfied resource offer. For our combinatorial auction, we implemented the CABOB algorithm (Sandholm, et al., 2005), which can solve combinatorial auctions involving hundreds of items and thousands of bids within a few seconds. Once the winner(s) have been determined, the auctioneer sends appropriate messages about the resources to be exchanged to the agents.

EXPERIMENTS

Experiment Setup

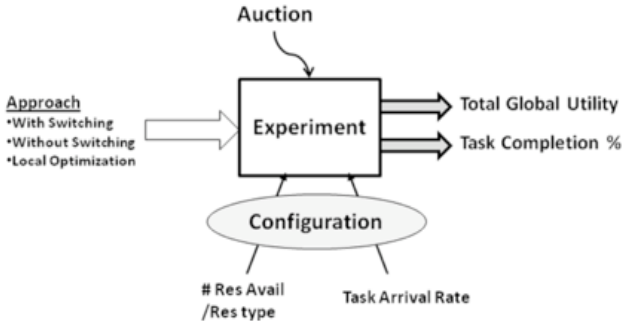


Figure 6. Experiment Setup

In our experiment, agents represented different relief organizations or teams and were assigned pre-defined tasks such as rescue, food delivery and medical emergency. There were a total of 100 tasks assigned to agents in the experiment and we a 100x100 simulated grid was the area of interest. Switching costs for each resource were based on the shortest distance from the current location of the resource to the requested location. While this is an extremely simplified switching cost function, in the future more realistic path planning and start/stop times can be included.

Figure 6 gives an overview of the experiment setup. The configuration control variables were *resource availability* and the *task arrival rate*. The *resource availability* of each resource type (i.e., helicopter, truck) was

varied from 1 to 3 to show the effects of relative amounts of resource scarcity on the results of the different approaches. All resources were assumed to have the same capabilities, reliability, and so forth. The only difference among each individual resource of a given resource type was its physical placement. *The task arrival rate* was varied in three levels (1 task every 5 seconds, 1 task every 10 seconds, or 1 task every 15 seconds). The *utility* value for each method of a task was generated based on a uniform distribution from 20 to 100.

Each experimental configuration was run for the three approaches described below and against our optimal allocation that over-estimates utility by setting the switching costs to zero. This effectively means that resources can be transported from one task team location to another instantly and without cost, which does result in the "optimal" allocation.

- *With Resource Switching* which is our approach
- *Without Resource Switching* which is an allocation without dynamic reallocation. In this case, agents are selfish and do not give up a resource being used. This can be compared to the Online Matching technique (Blum, et al., 2006) where a buy bid and sell bid are matched only if they are concurrent.
- A *local optimization technique* is a heuristic allocation with dynamic re-allocation where an agent locally evaluated the benefit of giving up a resource but did not have a global view of the system.

The experiment was repeated 20 times, each with a different set of 100 tasks and the results were averaged. The performance of the approaches was measured in terms of the total accumulated global utility and the number of successful tasks at the end of each run. The results are analyzed in the next section.

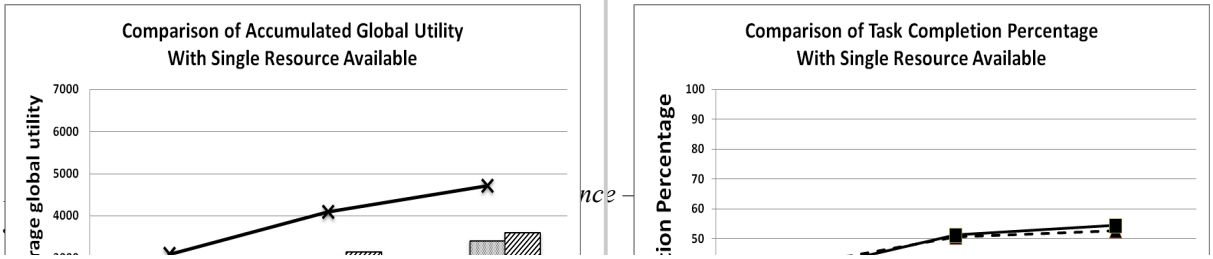
Experiment Results and Analysis

Figure 7 shows the comparison of the average accumulated global utility and Figure 8 shows the task completion percentage for different task arrival rates and the three different approaches for different resource availabilities.

From the results of Figure 7, we observe that our approach has a higher utility gain than either of the other two methods and is close to the optimal allocation when there are sufficient resources available and new task arrival rate is moderate. When resources are scarce, the global utility of our approach is slightly better than the no-switching approach and substantially better than the local optimization approach. Given severely limited resources, there is not much slack for switching resources around. However, both the switching and not switching approaches do better than the local optimal approach. Essentially these approaches do the best with the limited numbers of resources they have.

When both resources are scarce and tasks arrive rapidly, the global utility gained by our approach is only marginally better than either other approach. This low utility gain can be attributed to the tasks trying to switch resources in quick succession and losing utility because of the switching cost. To address this issue, we plan to incorporate the ability to evaluate expected utility as a function of time. As a task nears completion, its likelihood of success (and thus expected utility) would generally increase, making it less likely that these resources would be switched to another task. Nevertheless, as the task arrival rate increased, our approach did maintain the highest utility of the three approaches. The local optimization approach was not very effective when resource availability was limited but as the resource availability increased it performed slightly better than the no-switching approach. However, because of its local view, it was always outperformed by our switching approach.

From Figure 8, it is evident that the task completion rate is highest for our approach in most cases. Interestingly, when resources are scarce but task arrival rates are moderate, the task completion rate for our approach is very similar to that of the no-switching approach but our approach has a higher global utility. This occurs because our approach can focuses its limited resources on completing the higher priority tasks, providing for managed degradation under high stress conditions. Similarly, while the local optimization technique has a lower task completion percentage than the no-switching approach, it also manages to gain a higher global utility by shifting resources to complete higher priority tasks.



CONCLUSION

We have shown how market-based methods can be used effectively to coordinate distributed tasks and re-plan dynamically while reducing the need for elaborate agent reasoning mechanisms. By incorporating alternative resource reasoning over resource bundles, tasks can be dynamically adapted so as to maximize overall rescue mission utility while providing for graceful degradation under high stress conditions. We incorporate the notion of switching costs into our system to make our dynamic re-allocation both more principled and more real-world applicable.

Future work includes extending our simplified switching costs framework to consider detailed path planning considerations as well as more general switching constraints such as temporal and safety constraints. Longer-term investigations include market-based situation assessment. For example, if prices for helicopters remain high, then alerting procurement that more helicopters should be brought in. Situation assessment triggers could be useful for planning and training purposes as well.

REFERENCES

1. An, B., Miao, C., and Shen, Z. (2007) Market Based Resource Allocation with Incomplete Information, *Proceedings of IJCAI-2007*, Hyderabad, India.
2. Avasarala, V., Polavarapu, H., and Mullen, T. (2006) An Approximate Algorithm for Resource Allocation Using Combinatorial Auctions, *Proceedings of IAT-2006,* Washington, DC, 571-578.
3. Blum, A., Sandholm, T., and Zinkevich, M. (2006) Online algorithms for market clearing, *Journal of the ACM* , 53, 5, 845-879.
4. Blumrosen, L., Nisan, N., and Segal, I. (2007). Auctions with Severely Bounded Communication, *Journal of Artificial Intelligence Research*, 28, 233-266.
5. Boutilier, C. and Hoos, H.H (2001) Bidding Languages for Combinatorial Auctions, *Proceedings of IJCAI-2001,* 1211-1217.
6. Brenner, M. and Nebel, B. (2006) Continual planning and acting in dynamic multiagent environments, International symposium on Practical cognitive agents and robots, Perth, Australia.
7. Chevaleyre, Y., Dunne, P.E., Endriss, U., Lang, J., Lemaitre, M., Maudet, N., Padget, J., Phelps, S., Rodríguez-Aguilar, J.A., and Sousa, P. (2006) Issues in Multiagent Resource Allocation, *Informatica,* 30, 3–31.
8. Chevaleyre, Y., Endriss, U., and Lang, J. (2006) Expressive Power of Weighted Propositional Formulas for Cardinal Preference Modeling, *Proceedings of 10th International Conference on Principles of Knowldege Representation and Reasoning*, 145-152.
9. Cuena, J. and Ossowski, S. (1999) Distributed Models for Decision Support, *Multiagent Systems*, MIT Press.
10. Cushing, W., Benton, J. and Kambhampati, S. (2008) Replanning as a Deliberative Re-selection of Objectives, Arizona State University, CSE department.
11. de Weerdt, and Clement, B. J. (2009) Introduction to Planning in Multiagent Systems, *Multiagent and Grid Systems: An International Journal*, 5, 4, 1574-1702.
12. Fan, X. and Yen, J. (2007) R-CAST: Integrating Team Intelligence for Human-Centered Teamwork, *Proceedings of AAAI-2007*.
13. Fox, M., Gerevini, A., Long, D., and Serina, I. (2006) Plan stability: Replanning versus plan repair, International Conference on AI Planning and Scheduling.
14. Jordan, J.S. (1982) The competitive process is informally efficient uniquely, *Journal of Economic Theory,* 28,1, 1-18.
15. Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A. and Shimada, S. (1999) RoboCup Rescue: Search and Rescue in Large-Scale Disasters as a Domain for Autonomous Agents Research, *IEEE International Conference on System, Man and Cybernetics,* Tokyo.
16. Klein, G. A. (1998) The recognition-primed decision model, *In Sources of Power: How People Make Decisions*, Chapter 3, 15–30, The MIT Press.
17. Malone, T. and Crowston, K. (1994) The Interdisciplinary Study of Coordination, ACM Computing Surveys, 26(1): 87-119.
18. Mendoza, B. and Vidal, J. M. (2007) Bidding Algorithms for a Distributed Combinatorial Auction, *Proceedings of AAMAS-2007*, Honolulu, Hawaii.

19. Miller, M. (2006) MALLET-A Multi-Agent Logic Language for Encoding Teamwork, IEEE Transactions on Knowledge and Data Eng., 18, 1, 123-138.
20. Mita, H., et al. (2000) Solving an Escape Problem with Generic Programming (Japanese), *Proceedings of MACC2000.*
21. Nair, R., Ito, T., Tambe, M., and Marsella, S. (2001) Task Allocation in the RoboCup Rescue Simulation Domain: A Short Note, RoboCup.
22. Sandholm, T. (2002) Algorithm for optimal winner determination in combinatorial auctions, *Artificial Intelligence*, 135, 1-2, 1-54.
23. Sandholm, T., Suri, S., Gilpin, A., and Levine, D. (2005) CABOB: A Fast Optimal Algorithm for Winner Determination in Combinatorial Auctions, Management Science: Special issue on Electronic Markets, 51, 3, 374-390.
24. Shneidman, J., Ng, C., Parkes, D.C., AuYoung, A., Snoeren, A.C., Vahdat, A., and Chun, B. (2005) Why Markets Could (But Don't Currently) Solve Resource Allocation Problems in Systems, *Proceedings of the 10th Conference on Hot Topics in Operating Systems*, Santa Fe, New Mexico.
25. Suarez, S., and López, B. (2006) Reverse Combinatorial Auctions for allocating Resources in Rescue Scenario, ICAPS Workshop on Constraint Satisfaction Techniques for Planning and Scheduling Problems, Cumbria, UK, 62-65.
26. van der Krogt, R. and de Weerdt. M. (2005) Coordination through Plan Repair, *Proceedings of the 4th Mexican International Conference on Artificial Intelligence*.