Task Allocation for Multi-Agent Systems in Dynamic Environments

(Doctoral Consortium)

James Parker
University of Minnesota, Minneapolis, USA
jparker@cs.umn.edu

ABSTRACT

Multi-agent systems are frequently used in real world applications, with an increasing amount of agents and complexity. In order to accomplish the goals of the system, agents must cooperate using limited communication and knowledge of the environment. Task allocation for multi-agent systems in itself is a difficult problem and when both the tasks and the environment are dynamic, a robust yet efficient coordination is required. My thesis explores the use coalition formation and complexity reducing mappings to accomplish task allocation for multi-agent systems in difficult environments.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

Keywords

Search and Rescue; Robot Teams

1. INTRODUCTION

Multi-agent systems are robust, flexible and efficient compared to complex single-agent systems at a cost of coordination requirements. Task allocation is well known to be NP-hard in multi-agent systems, leading to a variety of different approaches. Solutions can range from centralized auction based methods to nature inspired decentralized algorithms. Decentralized are often more desirable due to lower communication requirements and avoiding the single point of failure common to most centralized methods.

My work focuses on dynamic environments where the agents can be added to the system or fail. These types of environments also require task allocation with uncertainty as to the location, completion requirements and size of tasks, and agents must travel through unknown and possibly unsafe areas to reach the tasks. New tasks can also appear and old tasks can expire, which forces the task allocation to have a real-time solution. According to the categorization of [1], this domain consists of non-static, discrete tasks that are sharable but not divisible. I also assume agents are selfless, cooperative and share a common goal.

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013)*, Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.

2. RESEARCH CHALLENGES

Many robotic system strive to be deployed in real world situations, however more research is needed in order to efficiently implement multi-agent systems. In dynamic environments, where tasks are initially unknown and can change over time, it is impossible for agents to consistently find an optimal solution. Additionally, when a solution is found it is hard to determine its relative effectiveness.

This work is evaluated in the RoboCup Rescue Simulator [2], an open source project built for a large number of agents on a city scale map. The scenario is designed to simulate urban search and rescue and uses heterogeneous agents who have limited communication ability. The Istanbul map is shown in Figure 1 with agents as circles, roads as light colored and buildings as gray and multi-colored. The RoboCup Rescue Simulator gives the agents the map information of the city before the disaster, but once the simulation starts agents must explore and discover the location of tasks. RoboCup Rescue Simulator has a yearly competition which provides a rich set of data to verify an algorithm's efficiency.

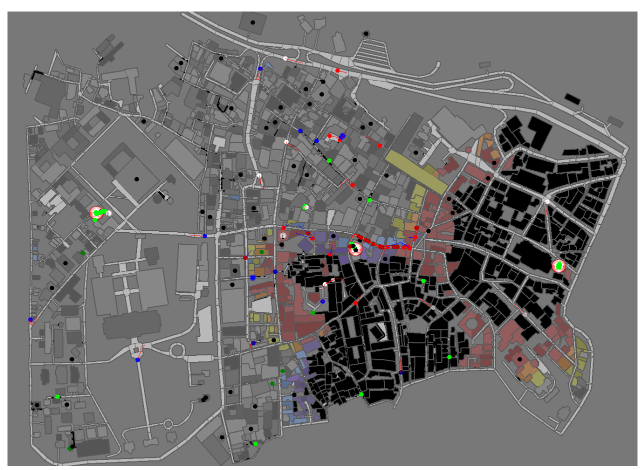

Figure 1: The RoboCup Rescue Istanbul map.

2.1 Forming long term teams

Nature is very unpredictable and harsh, yet life has flourished by exploiting the benefits of diversity. Humans use teams to create a cohesive unit that has a wider diversity and redundancy of skill, which increases the amount of tasks that can be accomplished while decreasing the risk of failure. Research in fully observable environments form coalitions for individual tasks, but after a task is completed the coalition is abandoned. Vig et al. [4] give a real-time solution in small-scale partially observable environments for a limited number of agents using auctions.

My first contribution is a more scalable approach using long term teams. Unlike coalitions which are normally formed temporarily for one task, long term teams are a group of agents that stay close to each other until all tasks are completed. This team formation more accurately corresponds to how humans generate teams, and ensures that there is sufficient resources nearby to complete any task found. Not only does this partition agents to reduce complexity, but also restricts space for each team to further simplify the problem.

Initially, teams are formed based on spacial locality using hierarchical clustering in order to reduce the amount of time agents spend moving to reach their teams. All agents on a team must maintain a close proximity to other agents, but the agents do not necessarily need to work on the same task. Agents within a team can use any algorithm that estimates rewards or utilities for actions to select tasks close to the team. Periodically each team shares a summary of the utilities of agents on its team with other teams. An agent with a low utility can transfer to a different team where similar agents have higher utilities if the utility lost when in transit can be quickly regained when at the new team.

The task selection algorithm, hierarchical clustering and agents transferring to different teams were all tested separately using the RoboCup Rescue Simulator. The basic task allocation on average slightly outperformed teams when just the hierarchical clustering was used but agents were not allowed to transfer teams. This showed that simply assigning agents to teams based on spacial locality was insufficient. However, when agents were allowed to transfer to different teams, the results were significantly better than the previous two experiments. After incorporating some domain knowledge to apply restrictions on team membership, this configuration ranked about third against the RoboCup Rescue Simulation League 2011 finalists using the same maps and configurations as the final round.

2.2 Modeling clustered tasks

Ramchurn et al. [3] describe a time sensitive task completion solution that uses coalitions with realistic constraints. Each task has a workload that must be reached before a deadline in order to be completed. I extend this problem to an environment where task workloads change over time and tasks can create new nearby tasks as time passes. Natural examples of this sort of problem include cancer, invasive species and forest fires. Instead of trying to deal with each individual task, my work focuses on clustering these tasks together and generating useful models that allow efficient solutions for task allocation.

Many task allocation problems become trivial if the travel time between tasks is zero, however when tasks grow over time this is no longer a simple problem. For a clustered tasks that grow under some assumptions[1], I show that there is an efficiently computable optimal solution when knowing all task clusters initially and with zero travel time. I then extend this to a real-time solution for a partially observable space in an algorithm within a bound of the optimal solution under a zero travel time assumption.

Unfortunately, with zero travel time the agents in the bounded real-time solution often reallocate to different task clusters for only minuscule gains. When travel time is non-zero there needs to be a dampening effect to minimize any thrashing of agent allocations. I give a non-zero travel time solution that utilizes a greedy heuristic to attempt and balance the task clusters so they all finish at the same time. This heuristic is based on trying to minimize the amount of time agents need to spend in transit when reallocation to a different task cluster.

The growth of task clusters in RoboCup Rescue was empirically derived to be approximately exponential, which satisfies the workload growth assumptions. After using domain knowledge to overcome the partial observability to estimate the size of the task cluster, the non-zero travel time solution was applied to RoboCup Rescue. This growing workload solution finished all tasks in 85% of the time as the long term teams configuration using domain knowledge.

[1]if $g(t)$ is the workload cost over time t then 1) task clusters can have different initial values but $g(t)$ identical for all clusters, 2)$\lim_{t\to 0^+} g(t) = 0$ and 3) $\frac{\partial^2}{\partial t^2} g(t) \geq 0$.

3. FUTURE WORK

In RoboCup Rescue agents and tasks are categorized into different types, where each type of agent can only solve one type of task, and agents within a type have the same amount of resources to complete tasks. Although the task allocation for an agent on a team is independent of this categorization, this classification is currently used to facilitate reallocations of agents to different teams. These categorizations allow agents of the same type to compare utilities and move agents with a low utility to a team where agents have a high utility. Since agents in RoboCup Rescue can only accomplish one type of tasks, agents on the same team can have drastically different utilities based on the type of tasks located near the team. More research has to be done on how this framework can be applied when agents cannot be placed into categories.

The clustered task modeling solution can only be applied to the most important type of task in RoboCup Rescue, and the other tasks do not have growing workloads to fit the assumptions necessary. Unfortunately, the task cluster solution does not compute the utility individual of agents and simply compares the effects on the task clusters as a whole, which does not allow this solution to be adapted into the long term teams. More work must be done to find a mapping which allows these approaches to be integrated.

4. REFERENCES

[1] Y. Chevaleyre, P. E. Dunne, U. Endriss, J. me Lang, M. Lemaître, N. Maudet, J. Padget, S. Phelps, J. A. Rodríguez-aguilar, and P. Sousa. Issues in multiagent resource allocation. *Informatica*, 30:3–31, 2006.

[2] H. Kitano and S. Tadokoro. RoboCup rescue: A grand challenge for multiagent and intelligent systems. *AI Magazine*, 22(1):39–52, 2001.

[3] S. D. Ramchurn, M. Polukarov, A. Farinelli, C. Truong, and N. R. Jennings. Coalition formation with spatial and temporal constraints. In *Proc. Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, pages 1181–1188, 2010.

[4] L. Vig and J. A. Adams. Coalition formation: From software agents to robots. *Journal of Intelligent Robotic Systems*, 50(1):85–118, 2007.