A model for Large-scale Team Formation for a Disaster Rescue Problem

Balaji Viswanathan and Marie desJardins
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore MD 21250 USA
{balaj1, mariedj}@cs.umbc.edu

Abstract

We present a model for multi-agent team formation and task allocation through implicit communication among the agents. Intelligent stationary agents (beacons) that are placed dynamically over the task area mediate the communication process among the task agents. Task specialization and the current situation in a region are used to guide the task allocation, team formation is based on the strength and type of known tasks. By utilizing decentralized communication and control mechanisms, inspired by biological swarming behaviors, very large teams can be formed in real time. We illustrate our approach in an application that requires the coordination of multiple tasks in a rescue operation for a tsunami scenario, in which the environment changes dramatically and very little prediction of task requirements can be done in advance.

1. Introduction

In realistic dynamic environments, very little information is available to allocate teams of heterogeneous agents to complete a group of tasks within a reasonable time scale. We use disaster rescue as a domain to implement our model for coordination among a large number of agents with different capabilities. Our goal is to generalize the model for similar domains, such as space exploration [12] and urban warfare.

RoboCup Rescue is a simulated environment for disaster planning involving multiple agents [5]. The environment is generally an urban location affected by some disaster such as fire or earthquake [3], the goal is for a group of heterogeneous agents to cooperate in performing tasks such as putting out fire, rescuing civilians, and clearing roads. The problem is interesting, not only for its potential for reducing humanitarian damage in an actual physical implementation, but also for the challenges it presents in terms of forming dynamic teams of heterogeneous agents in a fast-changing environment, where very little communication is possible among the agents. Although, many techniques have been applied to Robocup Rescue [1, 3], most of the approaches do not scale for very large teams [11].

In this paper we introduce the concept of beacons, which are intelligent stationary agents that are dynamically placed in the task region by the mobile agents. Each beacon coordinates the goal attainment in its local region, by mediating communication among the mobile agents that perform the assigned tasks. The beacons are assumed to have a relatively short communication range, and the mobile agents can only communicate with the agents, in their region. The mobile agents keep the stationary agents updated of the various task requirements in that region. This procedure is scalable, since it involves decentralized control by the stationary agents, and it allows effective coordination strategies to be formed among the agents in real time.

2. Related Work

RoboCup Rescue has been the subject of a number of practical implementations [3, 4, 5]. Usage of task allocation in distributed mobile sensor network is discussed by Low et al. [7] and the usage of network beacons for mobile robot exploration is discussed by Batalin et al. [2]. Ahmadi el al. [1] have used advice taking based approach to coordinate between multiple agents, in RoboCup environment. Architectures for large-scale team formation is a subject of current research [6, 8, 9, 10].

Batalin et al. [2] have used beacons in a sensor and communication network and have used robots to dynamically deploy the agents on the domain. However, the beacons used for their application do not involve in team formation among agents, and is used just an aid for navigation. Our approach involves the use of network beacons, with the beacons being autonomous and intelligent, guiding the task performing agents. In this model, we use the beacons that are dynamically deployed in the task domain as an agent for decentralized control and coordination.

3. The Task

We take our task to be rescuing humans from wreckage caused by a tsunami. The recent South Asian tsunami showed that damage can be widespread and civilians could be trapped in hundreds of damaged buildings in the task area. The goal of the agents is to rescue as many humans as possible in the shortest time range. The number of civilians may be in the hundreds and could be dispersed widely in the task region and each of them would constitute a rescue operation. Individual rescue operations might require the cooperation of multiple agents, the number of agents required for a rescue operation depends on the degree of damage, which has to be dynamically determined by the agents. Communication channels may be limited since many of the signal towers in the region could have collapsed, and the overall operation could potentially involve even thousands of agents, precluding the possibility of centralized control.

4. System Architecture

We have designed a general architecture that is applicable to many domains involving the coordination of a large number of mobile agents, where each agent possesses some task specialization, performing complicated tasks in a dynamic environment. Examples for this may include space exploration, wide area search munitions [8], urban warfare, and RoboCup Rescue.

4.1. Agent Structure

Our model involves the following agents:

1. Heterogeneous mobile agents, or task agents, that must coordinate to perform tasks that arise in the environment.
2. Stationary agents, or beacons, that guide the task agents in a particular region. The mobile agents have limited communication capability. They can only exchange data with a stationary agent that is within its signal vicinity. Task agents have the capability to move within the task area; for simplicity we assume that they can compute their current position using dead reckoning. The overall model is shown in Figure 1.

In the tsunami domain, the mobile agents are subdivided into:

1. General search agents that possess general-purpose sensors to detect humans, damaged buildings, and vehicles, and that can communicate with the stationary agents about the presence of targets in the region, with approximate positional information.
2. Task confirmation agents that possess precise sensors to estimate the degree of destruction and the number of humans trapped at a particular location. They calculate the number and type of rescue agents required for the operation. The stationary agents guide the task confirmation agents regarding the approximate position of targets to look for.
3. Rescue agents that use the predictions generated by the task confirmation agents to form teams with the necessary capabilities. After adequate team strength is attained, the agents move to rescue the target.
4. Deployment agents that possess a number of beacons; they navigate through the task area and use a simple criterion for deciding when to drop the beacons. If at any point they aren't receiving a broadcast message from any beacon, they deploy a beacon (stationary agent) initialized with the current time-stamp and positional information; otherwise they continue moving.The stationary agents consist of a network beacon with computational and communication capabilities that can be deployed in the task region. The stationary agents broadcast their signals over a short range, communicating with the task agents that respond to the broadcast message.

4.2. Team Formation

Teams are formed through implicit communication between the task agents, mediated by the beacons. The task domain is dynamically split into a number of smaller task regions, where each region has a guiding beacon that maintains information about the region. The information is collected through the communication between the beacons and mobile agents that visit the region. Figure 1 illustrates this process. The beacons also maintain an ordering relationship of tasks; when an agent enters its region, it tries to find a location where that agent might be required to perform the subtask in which it specializes. Some of the tasks might involve more than one agent; in this case the beacon advises the agents to wait for a sufficient number of the specified type of agents to gather. Thus, we have decentralized controllers that possess knowledge of the local environment and use this knowledge to engage the task agents in teams.

The task agents, on entering a region, look for tasks that they specialize in. Search agents navigate within the region looking for targets; when it finds one, it communicates with the beacon to register the presence of the target. The beacon then waits for a confirmation agent that can estimate the strength and position of that target. When a confirmation agent enters a region, the beacon guides the path to

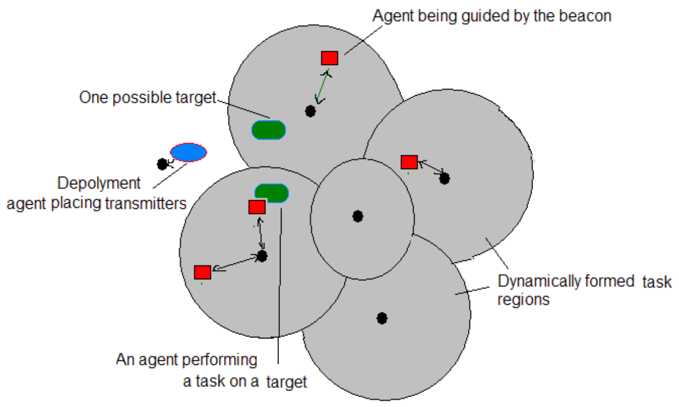
Agent being guided by the beacon
One possible target
Depolyment agent placing transmitters
Dynamically formed task regions
An agent performing a task on a target

Figure 1. A General illustration of the system interaction.

the approximate location of the target, and the confirmation agent sends information about the strength and location of target to the beacon. The strength value is used to estimate the number of rescue agents required and the beacons maintain the list of these estimates. When a rescue agent enters a region where there are targets to be rescued, it registers its presence, moves closer to the target and waits for signal from the beacon. Once sufficient rescue agents gather in that region, the beacon informs the agents to move to the target and the team of rescue agents performs the rescue task.

When an agent doesn't have a task assigned to it, it navigates around the environment, looking for targets in other regions. In the future work, we plan to work on enabling communication between the beacons, through the agents that cross from one region to another. This will allow the task agents to form teams more quickly, by information sharing among regions, forming dynamic groups of super-regions that maintain and exchange relevant information.

5. Results

We implemented the system in C++, modeling the agents as class objects. The physical domain, the mobile agents, and the stationary agents were implemented as separate classes that are monitored by a top-level controller class, which controls the simulation process. Targets were randomly placed in the task domain; all of the agents were deployed starting at one corner of the task region. The task domain was modeled as a N*M rectangular grid, and the agents could move one grid per simulation cycle. The agents have a randomized motion behavior; they persist with their current direction with a probability Pt determined by that type (rescue and confirmation agents have greater persistence, search and deployment agents have lower persistence); with probability 1 - Pt, they select a radom direction to move in.

The experiments were conducted by varying each of the parameters - number of mobile agents, size of the task area, and communication radius. Here, communication radius refers to the maximum signal propagation strength of the beacons; this range affects the size of the task regions, which in turn affects the team formation capabilities. The total number of beacons was set to be 1000. We used a specialization that assigned 15% of the total agents to be deployment agents, 35% search agents, 20% confirmation agents and 30% rescue agents. These values were chosen through a trial and error process to maximize the average performance for the domain. The number of targets was set at 50 in all cases. Since the distribution of the targets may affect performance, we performed 5 trials for each data set and averaged over them.

The main performance indicator in our case is the number of targets rescued within the allotted amount of time. Other performance indicators used were the distance traveled per target and the ratio of agents to the number of targets rescued.

In Figure 2, we compare the rescue rate for varying numbers of agents. We found that the rescue rate increased significantly as the number of agents increases. Here, we see that with 500 agents, 90% of the targets were rescued within 1500 time steps. In general, the number of targets rescued increases monotonically with the number of agents. This is also shown by the Figure 3,which shows the average number of targets rescued at a given time (t = 3000) increase, as the number of agents increases.

However, as shown in Figure 4, the efficiency of the agents fall, after reaching a certain threshold. In this case, we experimented with two communication limit values. The tests were run with a varying number of agents with communication limits of 50 and 90. The number of targets rescued per agent peaks around 100 agents for this particular scenario and then drops off. Another measure of efficiency - total distance traveled also provides a metric for comparison, since the energy spent, the probability of agent failure and positioning error all increase with increase in distance. In Figure 5 we see that the distance traveled per target increases with the number of agents. This shows that we need a trade-off between efficiency and rescue capability when deciding on the number of agents required for an operation.

Though there is an initial trend of increasing target rescue with higher communication access, this does not continue. This is because increasing the communication limits increases the degree of centralization,since each beacon controls a larger region, causing the performance to fall slightly after reaching a threshold. This is shown in Figure 6, where increasing the communication propagation, after a certain limit, slightly reduce the number of targets rescued. The graph shows that there was a performance gain when the communication limit was increased from 40 to 50. The performance, however, decreased when the communication limit was set to 60 and 100.

The total area of the task domain also affects the performance, since it determines the distance traveled by the agents and the size of the area to be searched. Figure 7 shows that increasing the task area slightly reduces the number of targets rescued for a given number of task agents. However, the reduction was not great indicating that the model will scale well to large task domains.

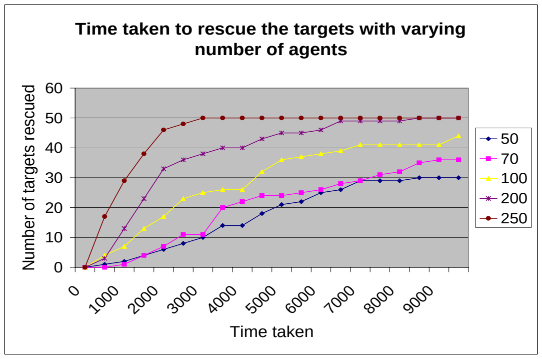

Figure 2. Influence of the number of agents on the time taken for rescue. The size of the task domain was 400 X 400; the communication limit was 50 units.

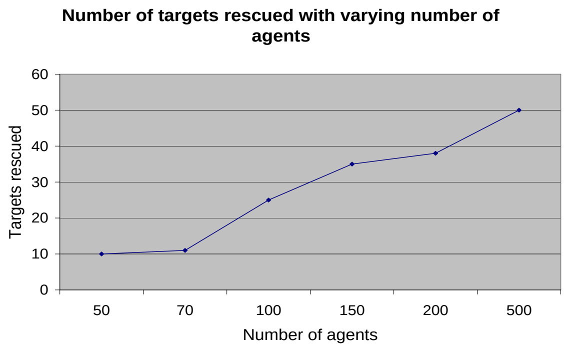

Figure 3. Number of targets rescued at t = 3000 with varying number of agents. The size of the task domain was 400 X 400; the communication limit was 50 units.

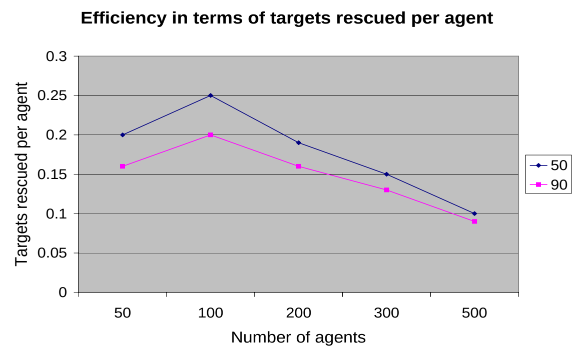

Figure 4. Efficiency of the agents in terms of targets rescued per agent, for different communication limits. The size of the task domain was 400 X 400; the time limit was t = 3000.

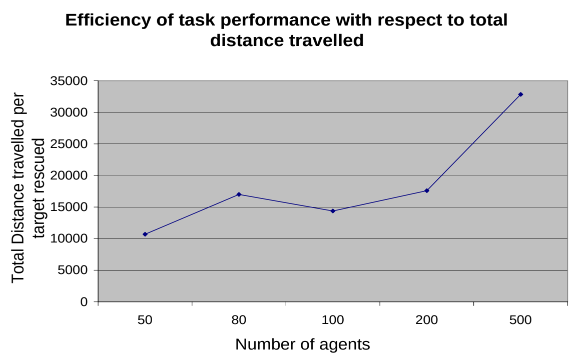

Figure 5. Efficiency of the agents in terms of total distance travelled per target rescued. Communication limit was 90 units; size of the task domain was 400 X 400; the time limit was t = 3000.

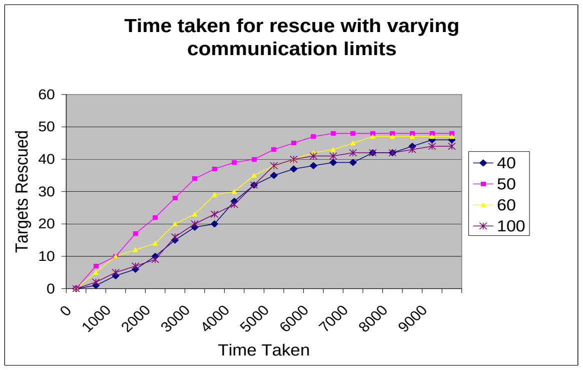

Figure 6. Influence of the communication limit on the time taken for rescue. The number of agents was 300, and the size of the task domain was 600 X 600.

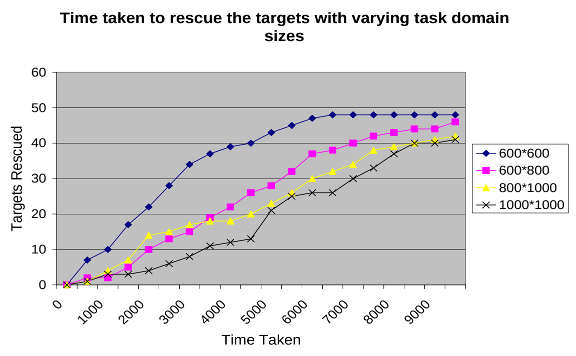

Figure 7. Influence of the task domain size on the time taken for rescue. The number of agents was 300; the communication limit was 50 units.

6. Conclusion and Scope for further work

In this initial work, we implemented a model for large-scale team formation using decentralized control provided by network beacons that are dynamically deployed in the task domain. This model was applied to a disaster rescue domain. Our experiments show that the model can be used for teams of large groups of heterogeneous agents, and that the efficiency of task agents, does not drop significantly as the number of agents increases. Task performance is also maintained fairly well, with an increased area of operation. We believe that this model will scale well and can be applied to a variety of applications.

In the future, we plan to integrate this model with existing RoboCup Rescue architectures and include more methodologies to compare our approach with existing techniques. We also plan to enable task weighting based on the time taken to rescue a target, which could guide the task allocation process. We are also exploring inter-region communication via mobile agents that cross region boundaries; this will create a communication network that will enable better transfer of information and increasing the efficiency of the system.

References

[1] M. Ahmadi, M. Motamed, and J. Habibi. Arian: A general architecture for advisable agents. In *MLMTA 2003: Proceedings of the International Conference on Machine Learning: Models, Technologies and Applications (MLMTA'03),June 23-26, 2003, Las Vegas, USA*, pages 17 – 23, 2003.

[2] M. Batalin and G. S. Sukhatme. Coverage, exploration and deployment by a mobile robot and communication network. *Telecommunication Systems Journal, Special Issue on Wireless Sensor Networks*, 26(2):181–196, 2004.

[3] J. Habibi, M. Ahmadi, A. Nouri, M. Sayyadian, and M. Nevisi. Utilizing different multiagent methods in robocuprescue simulation. In *Proceedings of the RoboCup-2002 Symposium, June 24-25, 2002, Fukuoka, Japan*, 2002.

[4] H. Kitano. Robocup rescue: A grand challenge for multiagent systems. In *ICMAS 2000: Proceedings of the Fourth International Conference on MultiAgent Systems (ICMAS-2000), July 10 - 12, 2000, Boston, USA*, pages 5–12. IEEE Computer Society, 2000.

[5] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takhasi, A. Shinjoh, and S. Shimada. Robocup rescue: search and rescue in large-scale disasters as a domain for autonomous agents research. In *IEEE SMC1999: Proceedings of the International Conference on Systems, Man, and Cybernetics, October 12-15, Tokyo, Japan*, volume 6, pages 739 – 743. IEEE, 1999.

[6] E. Liao, P. Scerri, and K. Sycara. Framework for very large teams. In *AAMAS'04 Workshop on Coalitions and Teams, Third International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS'04), July 19 - 23, 2004, New York City, USA*, 2004.

[7] K. H. Low, W. K. Leow, and M. Ang. Task allocation via self-organizing swarm coalitions in distributed mobile sensor network. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence, July 25-29, 2004, San Jose, California, USA*, pages 28–33. AAAI, 2004.

[8] P. Scerri, E. Liao, Y. Xu, M. Lewis, G. Lai, and K. Sycara. Coordinating very large groups of wide area search munitions. *Theory and Algorithms for Cooperative Systems*, 2005. Forthcoming.

[9] P. Scerri, Y. Xu, E. Liao, J. Lai, and K. Sycara. Scaling teamwork to very large teams. In *AAMAS 2004: Proceedings of the Third International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS'04), July 19 - 23, 2004, New York City, USA*, 2004.

[10] N. Schurr, S. Okamoto, R. Maheswaran, P. Scerri, and M. Tambe. Evolution of a teamwork model. *Cognitive Modeling and Multi-Agent Interactions*, 2005. Forthcoming.

[11] T. Takahashi, S. Tadokoro, M. Ohta, and N. Ito. Agent based approach in disaster rescue simulation - from test-bed of multiagent system to practical application. In *RoboCup 2001: Robot Soccer World Cup V*, pages 102–111. Springer-Verlag, 2002.

[12] S. Thakoor. Cooperative behaviors in small exploration systems. Final report - november 12, 1998, jpl d-16300a, Jet Propulsion Laboratory, Pasadena, California, 1998.