RoboAkut 2009 Rescue Simulation League Agent Team Description

H. Levent Akın, Ertan Doğrultan, Tekin Meriçli, and Ergin Özkucur

Department of Computer Engineering
Boğaziçi University, Istanbul, Turkey
{akin,ertan.dogrultan,tekin.mericli,ergin.ozkucur}@boun.edu.tr
http://robot.cmpe.boun.edu.tr/rescue

Abstract. RoboAKUT is a multi-agent rescue team developed at the Artificial Intelligence Lab of the Computer Engineering Department of Bogazici University. Our primary goal is to build a rescue team based on market paradigm and show that this method can be successfully implemented in highly dynamic, multi tasking and multi-robot environments. In addition to the general coordination issues we also work on multi-agent path planning which is not addressed by many of the teams.

1 Introduction

RoboCup Rescue Simulation environment is a disaster management simulation which consists of multi-tasking heterogeneous agents (Fire brigades, Fire Station, Police Forces, police Office, Ambulance Teams and Ambulance Center). In addition to being one the best test beds for agent coordination, there are many other challenges such as development of agent communication protocols for limited communication and delayed message arrivals, multi-agent path planning, scheduling, optimization, supervised learning for civilian death time and fire behavior estimation and unsupervised learning for agents to develop policies.

RoboAKUT is a multi-agent rescue team developed at the Artificial Intelligence Laboratory of the Department of Computer Engineering of Bogazii University. RoboAKUT performs rescue operations on the simulation environment provided by the RoboCup Rescue Simulation League. RoboAKUT has participated in the RoboCup Rescue Simulation Competitions held in Fukuoka, Japan in 2002, RoboCup 2003 held in Padua, Italy, Robocup 2004 in Lisbon, Portugal, Robocup 2005 in Osaka, Japan, Robocup 2006 in Bremen, Germany, Robocup 2007 in Atlanta, USA, and RoboCup2008 in Suzhou, China.

RoboAkut 2009 is a rescue team based on the market paradigm [1, 2] and we show that this method can be successfully implemented in highly dynamic, multi tasking and multi-robot environments. In addition to the general coordination issues we also work on multi-agent path planning which is not addressed by many of the teams.

2 Team Members and Their Contributions

– Ertan Doğrultan (Developer)

– Tekin Meriçli (Developer)
– Ergin Özkucur (Developer)
– H. Levent Akın (Advisor)

3 Key Contributions and Improvements Over 2008

The architecture of RoboAKUT is implemented from scratch for the 2009 competition. Most of the structure is new for the team. However, there are some modules that are used to maintain some specific tasks. The most important of those structures are the following:

– **Communication System** is basically a middle layer between the agents and the kernel to send/receive messages. This layer can be considered as a filter for each agent to get involved with the messages which they are related to. More importantly than that, this layer wraps the messages that will be sent to a specific channel to reduce the number of messages. This structure is explained in more detail in Section 5.
– **The Estimation Mechanism** is used to estimate the conditions of the civilians, the buildings and the roads on the map in order to allocate the agents more appropriately. This system includes fire condition estimation, civilian life span estimation, exploration and exploitation by using Kalman Filters. The Estimation Mechanism is still under development and we are planning to use this structure in competition 2009.
– **The Task Assignment Mechanism** is a revolutionary structure that is peculiar to RoboAKUT. An auction system is being used for the task assignment to each agent. This system varies for each agent type. More detailed information is included in Section 7. This algorithm is being added to the new system is a module.

4 Software Architecture

Our software system is based on *rescuecore* library.. The communication protocol and object types in the environment are provided to the system. The main modules are:

– **roboakut.communication :** provides enhanced priority management and message queuing functionalities. All types agents have specialized communicator objects,
– **roboakut.market :** implements market algorithms to the rescue environment,
– **roboakut.util :** has various functionalities. One important functionality is using log4j logger package for creating different debug logs for communication, task, message, sense activities, etc,
– **roboakut.task :** the main decision maker package. It includes various hierarchical actions specialized for ambulance, fire and police forces,
– **roboakut.firegrid :** provides geometrical calculations on burning buildings.

4.1 Layered Architecture

Layers define abstractions between the framework entities and minimize complexities due to coupled designs. Our framework is composed of three layers namely trading, scheduling and behavioral layers.

- **The trading layer** is responsible for auctions and relevant communication related to them.
- **The scheduling layer** is responsible for selecting the most profitable action in the task list, considering synchronizing issues like suspending or revoking of tasks. The assessments are necessary for dynamic environments where the state and the profitability of the tasks are rapidly changing. Assessment involves current task list and revenue cost models of the task in order to estimate the profit.
- **The behavioral layer** is concerned with robotic behaviors which are the primary means of executing tasks. The behaviors can be basic or complex. The robot's interaction with the world is achieved through behaviors.

All layers have access to message queue directly. However for environments in which the message bandwidth is limited the queue acts as a priority queue. The framework is depicted in Figure 1.

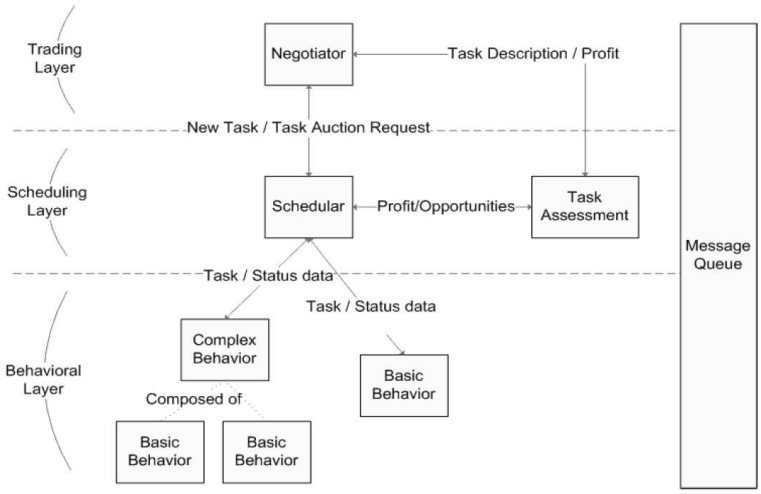

Fig. 1. Layered arch.

4.2 Software Tools

We do not have any in-house specialized visualization tools.

5 Communication

Communication is considered to be the most important part of our system to be optimized for saving as many civilians as possible. Sending radio signals from one agent to another agent directly requires a lot of processing time. A communication layer (See Figure 2) is implemented to make messaging more generic and pack/unpack the messages in one place. With the help of this system, the modularity of the structure increases and parsing the messages or putting the messages into priority order becomes much easier. This layer becomes the walkie-talkie of the agents that converts the radio signals into meaningful phrases or objects. The messages are sent to the kernel of the simulator and the kernel transmits those messages. Every message has to be a byte array to be processed. The Communication Manager class is a Serializable class to do this job. More explicitly, it takes strings, instances from an agent and converts it to byte arrays. Conversely, it processes the byte arrays to make them meaningful. The messages are not all controlled by the rescue agents. Since the ability to hear is also maintained by using the 0^{th} radio channel, there are also incoming messages from the civilians and those should also be processed.

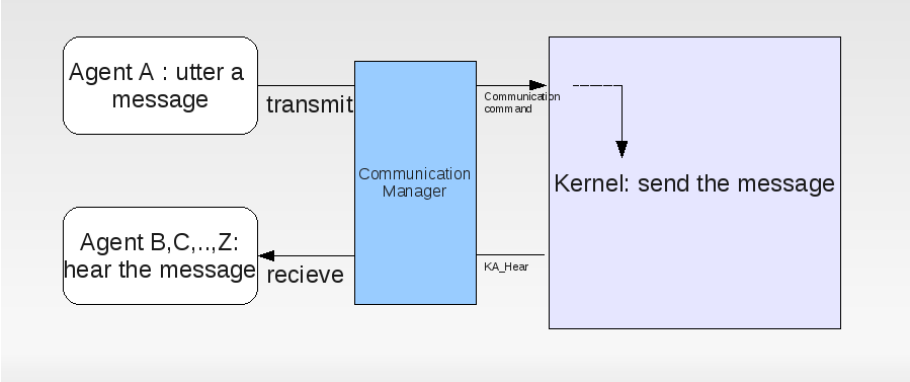

Fig. 2. The communication layer for the agents

In the first place, a message class that takes instances of *Humanoids, Buildings* or *Roads* is implemented with the communication manager. However, this is now seen as a mistake since this structure cannot include every kind of message to be sent within the agents. The message structure shall include every condition with many exceptions and this does not refer to just sending targets to the agents. Therefore, the message types are listed in detail:

5.1 Message Types to be sent within the Platoon Agents

Road messages

- Road is blocked

– Road is cleared

Building messages

– Building is on fire
– Building is extinguished

Civilian messages

– Position of the civilian that is seen
– Status of the civilian that is seen
– Civilian is rescued
– Civilian is heard (returns id)
– Civilian info (This type of message is the combination of position and the status of the civilian in case radio communication is not available)

Agent messages

– Agent needs help
– Agent is stuck
– Agent is busy
– Agent is available
– Agent needs to pass

These message types can also be extended for the special cases which means more detailed information can be transmitted in certain situations. For example, if one road is cleared by the police, only a prompt is generally enough. However, if this road plays an important part for a shortest path algorithm between two districts and the agent knows it, then it may specify the new path that will be a result of this action. The communication manager will also include a priority calculator for each message considering the evaluation of the competition. Depending on this priority setting and the type of the message, the agent decides if it shall process the message or not. The time complexity is vital in RoboCup Rescue Simulation. Hence, the agents should avoid processing redundant messages.

6 Search Algorithms

To take into account important factors like the blockades on a road, the crowdedness of the road, etc we use A* Search. Hence, the memory of the agent will be helpful to find a way between two locations since it is a form of weighted graph.

The basic heuristic i.e. the Manhattan distance between two points is used. It is obtained directly from a method of the Memory class. The $g(x)$ is the path cost from the starting point. The algorithm needs to return a sequence of *RescueObjects* elements that one *Road* object needs to follow one *Node* object. The structure does not accept any other kind of sequence. The *Building* objects may be a starting point or a destination, but they cannot be in the middle of the path. To be able to implement such structure, a *Node* class for searching is implemented. This *Node* class contains the path cost of

a node, a pointer to the parent of the node and the *RescueObject* that is represented by that class. The length of a *Road* is added to the path cost. On the other hand the nodes take their parents' path cost as theirs since it is for sure that the parent is going to be an instance of *Road* class. If the road has no free lanes or is blocked, the path cost increases proportionally. Hence, those roads are not favored while searching. A priority queue keeps the nodes to compare and order. In addition, the path cost to get to the goal is returned to the agent to determine if it is stuck or not. This is basically done by checking if that path greater than a specific value.

As mentioned in Section 5, when an agent hears a civilian shouting in pain, the exact location of the civilian is still unknown. The agent needs to search for the civilian and determine its position subsequently. A depth limited search that covers the 30 meters will be used to complete the search and find the hurt civilians.

The time complexity of the search is tested and one cycle is enough for finding many complex routes in many maps. Several additional heuristics will be added to increase the performance. Added to that, the police agents will be able to compare the paths and make sure that the vital ones are available. This can be determined for the known maps such as Kobe, Foligno and Virtual City (VC) before the simulation.

7 Agents

Currently RoboAKUT uses a multi-queue task scheduler for different priority tasks. The queue is divided into priority classes which are sub-queues of the main queue and each sub-queue is processed only after the more prior sub-queue is empty. A task first enters the queue with a default priority and the priority can change in time. For example aging is used for starvation prevention. In our implementation, our task planner implements a multi-queue scheduling where the priority of a task can be changed in time. Each task item in queue is prioritized according to other task items of the same type. Therefore default priorities of each task queue and priority calculation for each task is very vital. Our schema enables concurrent execution of different priority task while optimizing important tasks. This simplifies the agent design and priority calculation.

We use two different market algorithms for specific tasks according to the number of task items that should be completed. Rarely occurring tasks such as helping other agents by police force, rescuing civilians that can be saved by only one ambulance team, selecting fire region are auctioned by standard single item auction scheme. We allow only a limited re-planning for these tasks by only allowing the agents to auction for a high priority task while executing a task. Our new algorithm is used for task types which have numerous items such as exploration of buildings, block clearance and route checking where we have to optimize the execution of tasks since single item auctions cannot yield optimum solutions. In our approach we use the market algorithm where the tasks are bid according to a plan. The agent can bid if it is not assigned to any task of that kind. It will do some lower priority tasks and participate in other auctions if available. Since the communication between agents is limited within the communication hierarchy, the number and length of messages, negotiating very quickly by using a small band-with is a very important consideration. According to our paradigm the leader is the one who can minimize the communication. At each time step, each agent sends its status to no-

tify its center. The message includes the position of the agent and the assigned task type. This message is only heard by the center of the agent. The center collects these messages and issues a task list message for each task type that is coordinated by the market mechanism. For task types which heterogeneous agents are coordinated, centers exchange the task lists before sending it to the platoon agents. Upon receiving the task list the agent creates a plan by internally simulating the auction phase for each agent which is currently assigned to the same task. The agent selects the best target for him and auctions the task among his teammates and develops a plan for specified size. If the plan size is small all the agents will participate in the task, on the other hand if the plan size is large most of the task will be assigned to agent with larger plans and some of the agents cannot get a job. Therefore the size of the task plans controls the number of agents participating in the task and the total completion time. An indirect effect of this parameter is the level of multitasking where the agent performs different task types in specified time. Currently our agents do not perform a task if it can be completed by another agent and switch to low priority tasks which is not usually possible in multi-queue scheduling with predefined priorities. In case of communication failure the agent behaves as regions exists.

Platoon agents and center agents both implement the same auctioneer role to control the task assignment. Our preference is using the center however without implementing the role for ambulance teams the system is subject to single point of failure because of the ambulance center. The platoon agents leave the auctioneer role to the center if they detect the failure of the center or even it does not exist.

Since the auction mechanism is based on messages, due to message delivery time our agents start auction process before finishing the current task. For example, if the auction is among the same type agents, in the first step an agent opens the auction, in the second step all agents who want to participate in the task bid. Depending on the communication method implementation the agents can clear the auction themselves if messages are broadcasted or the auctioneer clears the auction and awards the winner(s) in unicast/multicast messaging cases. If the agent is idle it selects to operate the task first while participating in the auction.

7.1 Ambulance Team and Ambulance Center

Rescue Civilians Civilian information is acquired by the auctioneer. The civilian is categorized as *helpless, will die soon, will die* and *will live* according to time to live estimation and closeness to disaster area. For the civilian classification, civilian data dimension is decreased to one using PCA and classified using a Bayesian classifier. The life time estimation is based 4^{th} order polynomial fit of the PC data. A civilian which cannot be rescued even if all the agents participate in the operation is classified as helpless and is neglected. The will live and will die category are not taken into account unless all the buildings are explored. The number of agents needed for rescue operation is estimated and required number of ambulances is assigned the task according to simple auction.

Help Agent Helping an agent is the highest priority task. However if the ambulance is rescuing a civilian which will die soon and the agent is not under life threat the action

is postponed until the current task finishes. No auction is held and all ambulance teams are directed to the help waiting agent.

7.2 Police Force and Police Center

Clear Route Clear route requests are initiated by platoon agents when they are notable to reach a target. The message includes the last position of the agent (AP) and the target (TP). The clear route task must be completed immediately after it is issued however the time can only be minimized by assigning more than one agent to the task. The path is decomposed into long roads and auctioned separately. Therefore we implement a task decomposition scheme based on the map information. The long roads are also prioritized with Floyd algorithm.

Clear Blockade Clear blockade is a low priority task therefore the bidding schema designed to utilize less agent to the task compared to other police tasks. The cost for the blockade is the travel and clearance cost. In accordance to our planning paradigm the agent bids for the blockade according to its plan. As we previously mentioned, this approach decreases the total cost by assigning most of the tasks to one of the agents if the targets are very close. Therefore our agents have time to do other tasks. The target is selected by Floyd algorithm based priority, notified count and the count of clear route tasks which the block is included.

Help Agent. The highest priority task for the police force is assigned by single bid auction to the closest agent who is not executing a help agent operation. The bid is the time needed to rescue the agent.

7.3 Fire Brigade and Fire Station

Extinguish Buildings. Fires are very hard to extinguish which need effective collaboration among fire brigades. The first problem is to find the right building to extinguish. In our implementation we define a fire risk map of the environment. The environment is divided into small regions and two variables from threat and to threat are calculated for each time step. A region at the corner of the map is less dangerous than a region at the center because it threats less area. A fire threats its neighbors more than far regions of the environment. The total from threat is the measure of the region threat to the environment. On the other side, the total threat that affects the region is called to threat. The target building is selected first by selecting the target region which is the minimum threatened region that threats the other regions highly. The buildings that are likely to be extinguished easily which reside in the region boundary are selected as target. The agents are distributed among regions by single auction method. After a fire brigade is assigned to a region, it is free to select the best building to extinguish because the center is updated with a delay so it is useful for global planning not for specific actions taken locally. Fire brigade selects the building at the edge of the fire region which has the most probability of successful extinguish.

7.4 Common Tasks

Exploration. Exploration of the buildings is very important for a successful rescue operation. The civilians are injured and buried in their houses but some of them are vital

whereas some are not. A good exploration should find all vital damaged civilians before they die. The only way is to explore as many buildings as possible in the short time. Since the exploration is a common task it is coordinated among all agents by market with plan coordination schema. In case of communication failures, the map is divided into regions and the each region is assigned to at least one agent. The buildings are ordered according to their priority which is a function of civilian existence probability which increases by the number of civilians found in the vicinity, closeness to disaster areas. **Explore Region.** Explore region is very similar to explore unvisited buildings however it tries to maximize the traveled path in contrast to explore buildings. This task works only region based and has a low priority. **Check Civilians.** Although we apply machine learning techniques to estimate the remaining life of a civilian it is better to collect more data for a civilian. This task works only region based and agent tries to maximize the last updated time of a civilian.

7.5 Agent Skills and Action Selection

Multi-agent Coordination Although multi-agent systems have been developed for solving different types of problems, these problems share some common characteristics. Gerkey and Mataric [3] developed a taxonomy based on three attributes of the problem definition. Here after we will refer to this taxonomy as GM taxonomy. First, the problem is categorized as either single robot (SR) or multi robot (MR) depending on whether the task can be achieved by one or more robots. The next categorization is done regarding whether a robot is capable of only a single task (ST) or more (MT). Finally, the problem is categorized as instantaneous (IA) if all the tasks are known by the agents initially. The counterpart definition of the assignment property is time extended (TA) which describes the problems where the tasks are discovered during the course of action. These definitions are useful in classifying the problems in the multi-agent domain.

Both types of task assignment are available in the rescue simulation environment, since the map is known by the agents. Exploring the whole map is a IA type of task and since the blocked roads, fires and civilians are discovered during the simulation the tasks related to them are TA tasks.

We propose that by taking a simple plan into account while bidding in auctions, the agent is capable of exchanging multiple items in single item auctions. The proposed approach [2]is implemented mainly in the multi-robot exploration task where centralized solutions do not satisfy the communication and robustness requirements. In such solutions, all the agents communicate with the center which introduces the single point of failure problem. Moreover, if communication with the center fails or is noisy, the performance degrades sharply even to non-functioning level. The exploration task, however, must be completed in any condition even when only one robot survives.

In our approach, the agent simply plans a route that covers all the known targets by using the TSP insertion heuristic. Each agent auctions for the lowest cost target which is in fact the closest target to the agent. Other agents bid in the auction according to their plan cost. The plan cost is the distance between the agent and target if the target is the closest target or the distance to the previous target in the plan. The pseudo code for the algorithm is given in Figure 3.

```
check whether the target is reached
plan current tasks
bid for the lowest cost item
if auction won
        allocate task
else if deadlock detected,
        solve according to total plan cost
else
        stay
```

Fig. 3. Market plan algorithm pseudo code

Task Representation

Task definition is the key for a successful framework because it forms the basis of agent abilities in the market. The properties of a task in our framework are discussed in the following sections.

Task Constraints. There are several resource constraints in a system due to the fact that there are many different objectives to maximize utility of many resources such as time and fuel used by the agents. For the case of heterogeneous robots, robot types and capabilities are also a kind of resource constraint. In addition to the resources of a multi-agent system there are also coordination and synchronization constraints for tasks. The tasks may depend on each other's completion, the start and/or end time of a task. We propose task decomposition as a graph that represents these properties. Moreover we define task milestones for even atomic tasks when necessary. Task decomposition is used by police force agents to sell their tasks to other agents to minimize the total time of path clearance.

Task Milestones A task can depend on the completion of another task. Therefore task dependencies prevent the agents from starting a task before the depended tasks are totally completed. This preventing the start of a task is frequently encountered. For example, assume that one of the agents is occupying a grid and will use it for some time. Another agent, however, needs to use this grid to pass, but it knows that it is occupied for some time. In this case, the agent should be allowed to move along the path at least up to a destination which we call milestone. There are different kinds of milestones during the course of the task and the start is only one of them.

Task Opportunities Since the plans consist of the best task orderings, we can say that they eliminate the need for combinatorial auctions and put a kind of agreement which is calculated to be the optimum way of doing the job. However the IT cases cannot be handled because these domains do not allow future planning of the tasks because of the dynamicity of the environment. The problem can be solved by combinatorial auctions if it is ST-SR-TA or ST-SR-IA and task decomposition if it is a ST-SR-TA.

In our approach, for each task type, we propose to define a list of possible task types that can be executed simultaneously. Before bidding, the agent checks its task list and tries to find a task opportunity that can increase its revenue, i.e. decrease the cost of taking a task. The benefit of this approach is the same as task decomposition which

is trading for a task where the combination of tasks is planned behind the scenes and implementing such a scheme without using complex task models in the framework.

The default task opportunity for a task type is itself. Therefore task opportunities are the common version of the plan based bids paradigm. However its implementation is easier since each task type is encapsulated and complexity is reduced. For example, a police force can remove other blockades on its path to a specific blockade target. In contrast, an agent can explore buildings on the way to the blockade and while bidding it can offer less cost to the system.

Cost and Revenue Each task has a cost based on the resource requirements. However having representations that depend on only the cost is not sufficient for multitasking environments, since an agent may minimize its costs but this does not yield maximization of the overall profit. Therefore the revenue for each task must also be defined and used by the system. In a market economy every asset has its price and the price is determined by the market mechanisms like supply and demand. So the price is not static and it is also subject to change according to the changes in the environment. Therefore revenue is not static. In multitasking dynamic environments where tasks are assigned in time extended fashion any task can be abandoned if its cost is affordable. As a result, the cost is not only a function of the resource usage but also the opportunities that the agent is missing.

Task Notifications All the tasks the agent can execute or can be a part of are stored by all agents. Therefore no task is left unexecuted because the task is never lost due to robot failures or communication errors. But the agents cannot store the tasks forever even if the task is finished or expired. The task status also needs to be synchronized. Task notifications are defined for each task and sent when necessary.

Task Execution Preconditions of a task is valid for some time frame, a world state or simply exists in the environment. These preconditions must be checked for each time step. If an inconsistency is discovered other agents are notified as described above.

Task Allocation and Coalition Formation

Coalitions are necessary for most of the heterogeneous agent domains. Because of the diversity of the robot capabilities, typically one agent is not capable of accomplishing a task alone. Tasks can be categorized as ST when one agent is able to achieve it or MT when a coalition needs to be formed.

- ST case: The task is auctioned and one of the agents wins the auction. After this, if the task is decomposable the winner agent starts auctions for sub-tasks. A coalition is formed if any part of the task is sold.
- MT case: The tasks are auctioned depending on the task graph and a single task is reached. Then the subtasks are treated like the single task case.

Both cases are almost identical and show that the tasks can be decomposed and auctioned iteratively. For ST tasks there is another case in which the coalition emerges from the needs of the heterogeneous agent teams. The task types are predefined as ST or MT but while executing, a ST task that must be done by other types of agents can be discovered that blocks the agent and prevents it from doing the task. For example, while trying to reach a target, a block can be discovered on the road and unluckily on the only path to the target. In this case, the agent requests help from other agents by auctioning

as in the two cases. We call this dynamic coalition formation and the required task is added to the task graph of the requiring task.

Roles

The agents prepare auctions, bid for the auctions and execute the tasks won. All these roles are general for auction based frameworks. An additional leader role is defined for our framework.

- **Leader:** The coordination of the auctions requires high quality and high bandwidth communication. If all the agents are allowed to trade as they want, communication becomes the bottleneck for the system, especially when the communication channel is limited. Therefore, we use a leader role to minimize the communication requirements by communicating the robots within the coalition and inform them accordingly and act as the interface of the coalition with regard to other agents. So the leader must exist in order to minimize inter and intra agent communication. For example, for communication types where the range is important the agent which can communicate with most of the agents is selected as the leader. Since the leader is the most knowledgeable agent it can also be used for centralized planning of small tasks. The leader may or may not execute a task and this is not a constraint for the leader.
- **Task Executer:** Task executer acts in a coalition and communicates with the leader for the changes in the revenue function and executes the task. Task executer can join any other auction to increase its revenue. All agents are task executers even when they are not assigned to any task since the default rest task is executed whenever the agent is idle for all agents and shows that the agent is alive.
- **Bidder:** Bidder is a hypothetical entity which is in fact used for representation purposes to ease the understanding of the framework. Since each agent is executing a task all the time there is definition of an idle agent. The bidder is the name of the task executer which is participating in an auction.
- **Auctioneer:** The auctioneer role manages the auction process for the agents. Just like the bidder, this role is also hypothetical because an agent can start an auction any time for a task which is not in the scope of the coalition.

References

1. Kose, H., K. Kaplan, Ç. Meriçli, U. Tatlıdede, and L. Akın, "Market-Driven Multi-Agent Collaboration in Robot Soccer Domain," in V. Kordic, A. Lazinica and M. Merdan (Eds.), *Cutting Edge Robotics*, pp.407-416, pIV pro literatur Verlag, 2005.
2. Tatlıdede, M. U. and H. L. Akın, "Planning for Bidding in Single Item Auctions," First International Workshop on, Agent Technology for Disaster Management (ATDM), pp.85-90, 2006.
3. Gerkey, B. and M. Mataric. "A formal analysis and taxonomy of task allocation in multi-robot systems," *International Journal of Robotic Research*, vol. 23, pp.939–954, 2004.