RoboCup 2013
Rescue Agent Simulation Competition
GUC ArtSapience Team Description Paper

Dina Helal, Ahmed Abouraya, Noha Khater, Mina Fahmy, Fadwa Sakr, Salma Osama and Abdullrahman Elhusseni

German University in Cairo, Cairo, Egypt,
dina.helal@guc.edu.eg, [ahmed.abou-raya, noha.khater, mina.adel, fadwa.sakr, salma.fathy, Abdullrahman.mohamed]@student.guc.edu.eg

Abstract

This paper describes the contribution of the GUC_ArtSapience team to the Rescue Agent Simulation; in terms of the current research approach implemented to prepare for the RoboCup 2013 competition. This year we are extending last year's approach which was modeling of the problem as a multi-agent planning problem. Task allocation is handled mainly through the use of clustering to divide the map among the agents. Coordination depends on the use of centers and communication, if available, but also can be done dynamically without the use of communication. This paper also outlines the differences from our approach in 2012, namely combining the advantages of using both fuzzy C-means and K-means++ clustering algorithms, enhancing our Ambulance Team implementation through tasks prioritization and using the RoboCup Rescue Simulation Communication library in our communication model and implementing different noise handling techniques.

1 Introduction

Rescue planning and optimization is one of the emerging fields in Artificial Intelligence and Multi-Agent Systems. The RoboCup Rescue Agent Simulation provides an interesting test bench for many algorithms and techniques in this field. The simulation environment provides challenging problems that combine optimization (routing, planning, scheduling) and multi-agent systems (coordination, communication, noisy or missing communication)[2].
The Robotics and Multi-Agent Systems (RMAS) research group at the German University in Cairo (GUC) was established in September 2010. The goal of the research group is to study and develop AI algorithms to solve problems in robotics and simulation systems. These fields include computational intelligence, constraint programming, computer vision, multi-agent systems, and classical AI approaches. The current research efforts investigate the following research directions:

- Clustering for task allocation and coordination.
- Using more than one clustering algorithm and combining their advantages.
- Investigating different noise handling techniques in the communication process.

- Tasks prioritization.

The GUC_ArtSapience team is making its third participation to the Rescue Agent Simulation in 2013. Our first participation (as RMAS ArtSapience) was in 2011 and ranked 3rd in the final round, our second participation was in 2012. This paper describes the current teams achievements in tackling the RAS problem. The remainder of this section summarizes the changes in our approach. Section 2 describes the changes in clustering in our approach and using shortest path distance versus the euclidean distance. Section 3 describes the enhancements we did in the ambulance agents' think and how they prioritize their tasks. Section 4 describes the changes in our communication model library and our noise handling techniques. And finally in section 5 we give some empirical results.

Changes from 2012

The following outlines the most important improvements made to the 2012 approach:

- Combining the advantages of using both C-means and K means ++ clustering algorithms.
- Measuring shortest path distance between entities instead of the euclidean distance when assigning targets to clusters
- Enhancing the ambulance task allocation.
- Implementing the RoboCup Rescue Simulation Communication library and integrating it in the code.
- Including noise handling techniques in the communication process.

2 Our approach

This section describes the modifications we did in our last year's approach to address the Agent Challenge. The first part discusses combining the advantages of using C-means and K-means++ clustering algorithms to divide the map into regions. The second part explains the changes done in our centralized communication model.

2.1 Task Allocation

The rescue problem can be divided into three main tasks (as previously explained in last year's team description paper)[1]: extinguishing fires, rescuing civilians, and clearing blocked roads. The previous tasks could be furthermore divided into several individual tasks.
Finding routes to buildings and refuges will be done individually by all agents. Extinguishing buildings can only be done by fire brigades. Rescuing buried civilians and

moving the injured ones can only be done by the ambulance teams. Clearing blockades can only be done by police forces.

2.1.1 Previous approach

In the rescue simulation environment, there are a lot of information that are obscure, for example, in disaster scenarios, the initial locations of fires, buried and injured civilians, and blockades are unknown. Moreover, tasks that are assigned to an agent don't specify where exactly that agent should carry out these tasks. So agents have no other choice but to traverse the whole map and search for them. And since it is impractical for a single agent to cover the whole city map (all buildings and/or roads), an approach was developed to divide the map into smaller parts.
Clustering was the method we chose for dividing the map. As a result each agent will be assigned to a region of the divided map, where it will traverse all of the buildings and roads in the region to search for events that require rescue actions. In 2011, fuzzy c-means clustering was used to generate overlapping clusters. The number of clusters was chosen such that each cluster would have more than one agent. However, it was clear after some testing that the best choice would be to have the number of clusters equal to the number of agents. Then there was the problem of extra computational time that this approach needs which will exceed the limited time allowed for preprocessing. In order not to sacrifice cluster quality, we chose to use K-means ++ clustering algorithm [3], which works similar to K-means, but uses a different heuristic for selecting the initial centroids. The initial centroids are selected from a uniform Gaussian distribution over the buildings in the map. This technique gives a much better start for the algorithm as well as guaranteeing a faster conversion to the optimal centroids.
Using K-means++ algorithm lead to overcoming the time issue and we were able to compute clusters equal to the number of agents within the allowed time. Even more, and since the K-means++ algorithm is faster than the fuzzy C-means algorithm, we were able to perform more computations during preprocessing such as calculating the shortest path distance between entities.

2.1.2 C-means versus K-means++

C-means clustering is a method of clustering that allows one piece of data to belong to one, two or more clusters, such that there are many parts in the map that exist in one or more tasks for an agent, the advantages of using C-means clustering is overcoming the fact that there exist blockades all over the map which might prevent agents from reaching their targets, by increasing the number of agents heading for this target, the probability of this target being covered increases, on the other hand this wastes time if the two agents head for the same target.
K-means++ same as k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, this method solved the problem of agents wasting time going after the same targets, however the problem of targets not being reached still exists.

Our approach this year is to verify the possibility of combining both advantages of K-means++ and C-means algorithms, the agents will modify their own clusters based on some factors after the simulation has been started. Before the simulation starts the agents do not know how the disaster looks like, how bad the roads are blocked or how many civilians are buried, So a possible way to overcome this problem is to use the K-means++ algorithm then after the non-overlapping clusters have been computed, each cluster will include regions of its other neighboring clusters while differentiating between the main tasks, and the common tasks at the borders of the region, after the simulation starts the agents could determine whether their region has a lot of blockades or not, in case the region is not blocked, agents should notify neighboring agents to remove common regions, otherwise the neighboring agents will by default explore their common task.

2.1.3 Factors affecting clustering

Some factors affecting the size of the common tasks are known before the simulation starts, among these factors the fact that the ratio between the size of the map and the number of agents of its own type is directly proportional to the size of the common task and the number of channels and their bandwidths is inversely proportional, so in case of having only voice communication all agents will explore common tasks if not other main tasks as well. For the Ambulance Team and the Fire Brigade the ratio between the size of the map and the number of police agents is inversely proportional to the size of the common task. After the simulation starts the Ambulance Team and the Fire Brigade know the density of blockades in their own clusters, which is directly proportional to the size of the common task, in our approach all agents tell whether they are buried or not, therefore the number of active agents could be calculated, which is inversely proportional to the size of the common task.

2.1.4 Shortest Path distance

In 2012 the Euclidean distance was used in the Kmeans++ algorithm to determine which centroid is closest to the target. The Euclidean distance is not really an indication of how far a target is from a centroid, and given the fact that Kmeans++ is fast, adding more computation to the algorithm is possible. The distance of the shortest path between the centroid and the target will be the measure instead.

As shown in figure 1 using the Euclidean distance the red building belongs to the green centroid, however using Shortest Path Distance the red building belongs to the yellow centroid which is more realistic, and in this case the agent assigned to this building requires less time steps to reach it.

3 Ambulance Team

The main task of the ambulance agents is finding buried civilians and rescuing them. This is one of the most important tasks in the rescue problem and it has major effects

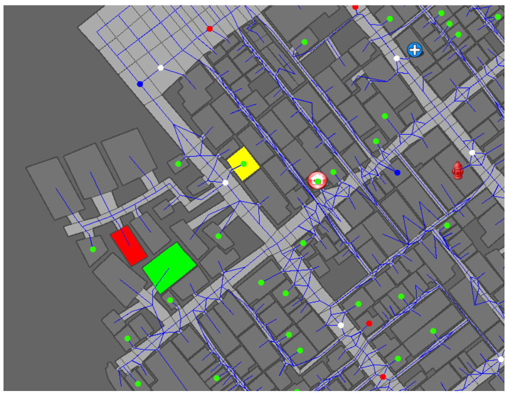

Figure 1: Shortest path distance vs. Euclidean distance

on the final score. Last year it was noticed that our weakest point was in rescuing the civilians. After some investigation we decided to modify the ambulance agent's tasks in a way that would increase their performance to rescue more civilians. Our main strategy to improve ambulance teams' performance is prioritizing their tasks. As explained in the previous section each agent is assigned to a cluster and each cluster has a set of targets that the agents start adding to their tasks' lists.

3.1 Tasks Prioritizing

Each agent has a list of tasks (targets) to rescue, the question is which one to choose first. The answer to that question depends on many factors such as the distance between the agent and the civilian, the civilian's health based on its buriedness state, whether the agents is in a burning building, near a blockade or not. Taking all these factors into consideration the ambulance agent is capable of making better decisions regarding which civilians to be saved first and if there are civilians that will die before the agent can reach them.
If an agent has only one task, it executes it regardless of any factors. The highest priority is always to saving buried agents. When the ambulance agent has many tasks (many civilians to be rescued), first it selects the closest civilian to it and checks its health and how buried it is, there some cases in which the civilian could be almost dead and can't be rescued, in that case the agent will leave it and move on to the next task in order not to waste time trying to save it while other civilians who could have been rescued earlier are getting worse. If the civilian is not dying and there is a chance to rescue it, the ambulance checks whether it is in a burning building or not, if it turns out to be in a burning building, it lowers its priority and moves on to another civilians, at the same time of course it reports the fire hoping that by the time it reaches that same task again the fire would be extinguished. If the road to the civilian is blocked, the agent will report that and also lower the priority of the task until the path is not blocked anymore. Prioritizing tasks saves a lot of time that was wasted on tasks that should be ignored or were being executed at the wrong time and needed to be done later. Postponing tasks

or lowering their priority in the cases mentioned above not only makes more time to save other civilians, it also prevents jeopardizing the agent who could die in a fire or get stuck in a blockade and be rendered useless.

4 Coordination after Planning and Communication

In 2012, the main scenarios that were considered included one with enabled communication and centers available, another with enabled communication but no centers, and finally, one without communication available. After planning, coordination was executed dynamically during the execution of the agents' plans. This strategy and approach is continued to be adopted in 2013; however, there have been some changes and new techniques introduced. These changes include implementing the 'Robocup Rescue Simulation Communication Library (RCRSCS)' and integrating it within our code, introducing noise handling techniques, in addition to some minor changes in the overall strategy. These changes and modifications will be described in details in the following sections.

4.1 Implementing the 'Robocup Rescue Simulation Communication Library' (RCRSCS):

The RCRSCS library provides the communication protocol between agents and center. The coordination structure of the RCRSCS library is based on a central coordinator to whom the responsibility of making decisions is delegated. The central coordinator receives all the available information known by the individuals from the environment and decides the tasks that ought to be performed by each individual in the disaster environment. The RCRSCS communication library has three predefined types of messages that could be sent: Information Messages, Task Messages and Report Messages. Information messages are those which represent information from the disaster environment. Task messages represent the orders which the platoon agents are given to perform. Report Messages simply report the order results of Task Messages that are sent to agents; the agent reports back whether the assigned task was accomplished or was unable to be completed.

The RCRSCS library is integrated into our code replacing the communication model and protocols that had been used during the past years. In 2011, our communication model had three types of messages: Informative Messages, Query Messages and Acknowledgments. In 2012, our model was expanded to include centers. All centers would receive information reported by the agents from the different clusters and continue to store such information about the world. Then, the centers would assign the free agents (agents who finished traversing their clusters) individual tasks according to the information obtained. Evidently, the coordination structure of the RCRSCS library is quite similar to that of our previously adopted model whether it is concerning the predefined types of communication messages or the reliance on a central coordinator (the centers). Therefore, the transition from the old model to the RCRSCS library model would not lead to drastic changes in terms of the code logic or the overall coordination strategy. On the contrary, it would aid the team in focusing on the strategy and would provide a broader, more comprehensive options for communication and coordination.

The decision to implement the RCRSCS library and integrate it in our code was due to the advantages of having more options when it comes to the types of messages that could be sent, thus contributing to the betterment of our strategy. In addition, the messages that could be sent do not include unnecessary data or information, thus reducing the size of the message sent compared to our old model. And consequently, the bandwidth of the communication channels in the communication enabled scenarios would be better utilized allowing for more efficient communication.

4.2 Communication Strategy and Approach:

For the communication strategy, the same task sharing approach that was used last year is still adopted and used. The agents and centers subscribe to the communication channels taking into consideration available number of radio channels, maximum number of channels an agent can subscribe to, and the maximum number of channels a center can subscribe to [1]. Different messages will be reported depending on each type of center. And the centers would prioritize the stored information so that it would handle the events based on their urgency[1]. Moreover, the behavior of the agents and centers in the case of the three main scenarios: communication with centers, communication without centers and communication-less scenarios.

The modification to the strategy that has been added this year is that the centers will be reporting information messages about the disaster environment. In the old model, the centers would store information reported and assign tasks to the free agents when they finish the tasks concerning their own clusters. However, now the centers get to act a more active role through sending Information Messages alongside its role in sending out Task Messages. This is achieved through the implementation of the RCRSCS communication library as explained in the previous section. The preliminary results have indicated that by making the centers report information, the scores in some of the maps have improved.

4.3 Noise Handling:

In the previous years, our approach did not include noise handling techniques. However, this year it is being taken into consideration, due to the heavy reliance on communication and as an attempt to make the communication process more efficient. In the communication channel, there are two types of noise: dropout noise and failure noise. Dropout noise is when the receiver receives a message without the content (content is lost), while failure noise is when the message disappears completely without notifying either of the sender or the receiver. In order to overcome the noise, the messages are sent more than once. First, the probability of noise in the radio channel is obtained. And according to that probability, the number of times the message is sent is determined. The receiver also checks for duplicates in order to avoid storing repeated redundant messages.

5 Results and Conclusion

The testing was carried out on the RoboCup Rescue 2012 maps and scenarios. Table1 shows a comparison of our current scores in some of the 2012 maps to the ones we got during the 2012 competition.

Table 1: Current scores compared to some 2012

Map	VC1	Kobe1	Berlin2	Paris2
2012 Score	105.204	61.801	27.9670	58.1520
Current Score	110.69	90.318	30.271	65.281

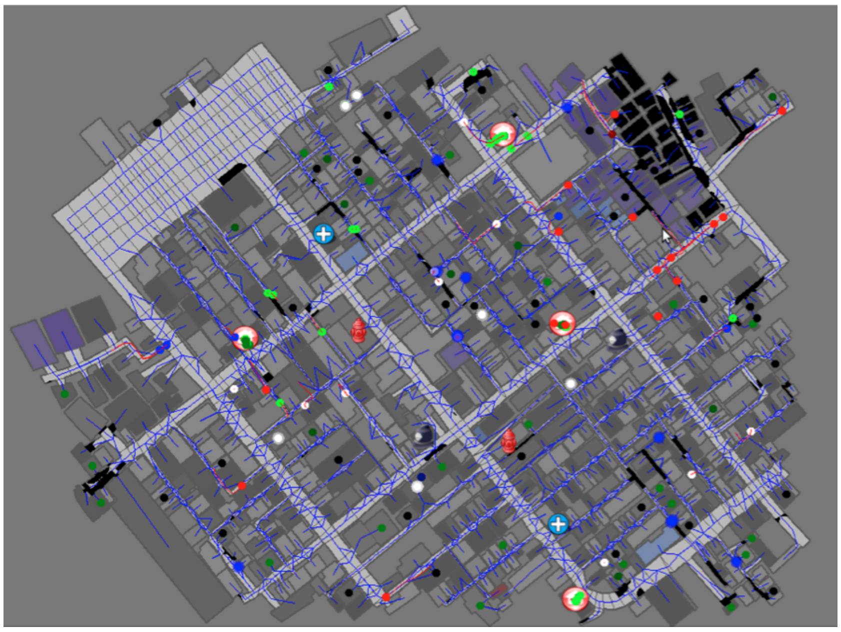

Figure 2: Final state of Kobe1 now.

The increase in the scores is mainly due to improving the ambulance team's performance using tasks prioritizing in addition to the improved clustering technique which resulted in a better distribution of the agents. The improvement in distribution increased the efficiency of the already implemented strategies in coordination with and without communication. Figure 2 shows our current state in map Kobe1 which was in 2012 day 1. As can be seen in the figure, the number of dead civilians and agents decreased and the fire was controlled. Further improvements are being tested in our approaches.

References

[1] Guc artsapience team description paper 2012.

[2] Robocup rescue website. http://sourceforge.net/apps/mediawiki/roborescue/.

[3] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. *Proceedings of the eighteenth annual ACMSIAM symposium on Discrete algorithms*, 8(2006-13):1027–1035, 2007.