Rayan Team Strategy Description

Mahdi Tashakori, Adrin Jalali, Mohammadreza Jooyandeh, Abdolaziz Gholami, Ali Behzadian Nejad, Kaveh Ghasemloo, and Behnam Esfahbod

Math and Computer Science Department,
Amirkabir University of Technology(Polytechnic), Tehran, Iran.
Sponsored by NAJA Research and Development Institute
(hashemi, adrin_jalali, mr_jooyandeh, gholami, behzadian, ghasemloo, behnam)@aut.ac.ir
http://www.fanux.org/rayan

Abstract. In this paper we will present the description of strategies and algorithms used by Rayan Rescue team participating in RoboCupRescue2004 in rescue simulation branch. We will present our base code, strategies and algorithms used for agents.

1 Introduction

Beside our interest in the programming and algorithmic aspects of the robocup rescue simulation, we are specially motivated to try to improve the algorithms for this task, with hope that these algorithms can be useful in the future situations in the real world. With this aim, our work has some main parts, we had some work on path planning, designing an efficient message protocol, and strategies to make agents, less center dependent; because there is the possibility of not having a center for some agent types, we tried to design agents such that they could work in lack of center. We also implemented a medium level message layer that uses agents of other kinds and their centers to make communication between centerless agents possible. Bellow is a brief explanation.

2 Base Code

Our base code is SOSs base code, that we greatly associate it. Two path-planning algorithms are implemented there : Dijkstra and focused D*. Using Dijkstra directly for maps isn't efficient. But we know that there are many nodes and edges in the map, that are not important in path finding at all. For instance when a node is neighbor of the only two roads (and some buildings), we can exclude it. After doing this, we have a graph that is not so large, such that the Dijkstra algorithm works enough fast on it. But it's not good when the changes of the disaster space is important for us (dynamic situation). For example police forces need to find the path very quickly, and change it according to the changes made on the disaster space. So we use focused D* algorithm for them. We have made

some changes though. For example, in these implementations they didn't take junctions into account. Junctions are important because when an agent reaches one of them and wants to turn, some time will be elapsed due to acceleration which is taken into account by traffic simulator. So having excluded junctions as more as possible, we reach the destination sooner.

3 Communication Protocol

We call any small part containing a meaningful information, an SMS. Each SMS has two parts. The type of the message and the body of that. For instance when an agent sends a building information, the type of the SMS will be BUILDING_INFO, and the body of the SMS contains a number showing the id of that building and its fieryness. We encode the id of objects such that we can send the encoded id only in two bytes. This is very useful because the length of the messages are restricted. Now a message packet which is sent, includes a number of SMS parts. Each SMS is either public or private. Public SMSs are messages that all agents understand, like civilian information which is sent to all agents. Private SMSs are allocated for agent-center and agent-agent communication. These include commands sent to agents from the center or information sent to center which are gathered by agents.

4 Agents

As mentioned above we tried to make agents work without a center in a given disaster space. For this purpose whenever the center does not exist, one of the platoon agents takes the responsibility. In this situation, the communication between different kinds of agents is possible only by means of "say command". We use it to transfer our messages through other agents and their centers.

4.1 Police Force and Police Office

The main part of police decisions is based on a function that estimates the score of the simulation. This function supposes that if the police force does a given task, other kinds of agents will do their tasks perfectly. For example, when a police opens the way to a fire point, supposes that fire brigades are able to extinguish the fire. So it can estimate the score of the simulation after doing a sequence of tasks. We know that the score depends on the burnt area and the hit point sum of the humanoids. Both of them depend on many parameters. In our implementation we try to include connectivity degree of the city graph, number of agents in each connected component of the graph, and many other less important details. We use a linear function of these parameters currently. The

coefficient of these parameters will be set manually now, but later on, they can be set by learning algorithms. For searching the job allocation space, our idea is to use taboo search or may be simulated annealing or may be a combination of them. Each police agent will search a restricted part of the space related to him, using these algorithms. It will send the results to the center, which is searching the solution space parallel with agents, but without their restrictions, center will take those results into account. by using this information, it will find a better local optimal job allocation space faster. At the end, when search is finished and tasks are assigned to the agents, commands will be sent to them. Depending on what agents are doing currently, we suppose a job for them. We categorized jobs into 5 groups:
1. Cleaning the path of fire brigades to the fires and refugees.
2. Cleaning the path of ambulances to the civilians and refugees.
3. Searching the city for probable changes to disaster space made by after shocks.
4. Cleaning around of fires.
5. Increasing the connectivity of the graph and decreasing the number of connected components of the graph.
We mostly focus on types 1 and 2 at the beginning. As time passes, we will put more overall time of agents to other works. We try to avoid changing job of an agent before it accomplishes its current job.

4.2 Fire Brigade and Fire Station

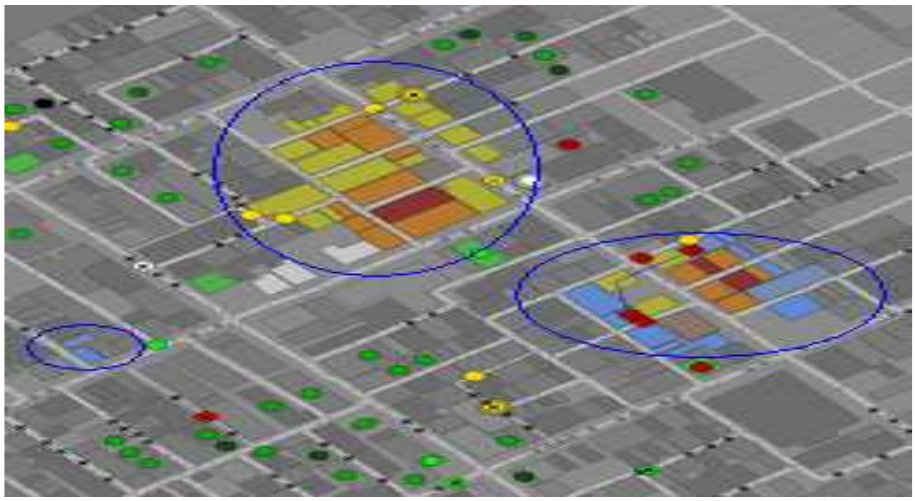

Fig. 1. Centers of fires assumed by fire brigades

At the beginning of simulation, some buildings begin to burn and when time elapses, neighbors of fiery buildings also begin to burn . So some regions of the

disaster space can be supposed to be centers of fire. The agents set some priorities for these centers according to some parameters such as the elapsed time from beginning of burning, number of civilians in that center, number of burning buildings, material of buildings and so on. Some agents (number of them depends on the above parameters) start to go to that center to extinguish. The order of buildings to be extinguished in the same center is very important. The permutation is obtained by using many different parameters. We try first to extinguish the around of center of fires. Because where a buildings burns and then due to elapse of time or extinguishing of fire brigades, its fieryness becomes higher than 3, it will not catch fire again. So fire will be controlled in this way. After controlling the fire, we start to extinguish which are inside the centers.

4.3 Ambulance Teams and Ambulance Center

Ambulance agents work in teams. Due to lack of information, about the situation of the civilians at the beginning of the simulation, they scan the city and gather information. Whenever enough information is gathered about the agents and civilians, they will be categorized. These categories contain several main parts:
1. Agents and civilians that are in urgent situation, which must be rescued as soon as possible.
2. Agents and civilians that are buried but aren't in urgent situations which can be rescued after a while.
3. Agents and civilians that are buried and can not be rescued. They will be ignored.
This categorizing depend on many parameters. Agents under collapsed buildings have the highest priority. Priority function also depend on h-point, buriedness, damage, the distance to the nearest ambulance agent,... is taken into account for each civilian. Agents update priority list through communicating each other. If an ambulance center is present, it will coordinate works of ambulances and assign their tasks. Center has a list of all known civilians. If any urgent interference is needed for some civilian, center will send messages to all of ambulance agents in order to rescue them quickly. In this case all of the agents must reach the target position at once, and rescue them. If there is no current need for an urgent rescue operation, agents divide into two groups to rescue civilians. If there is no need for a rescue operation, agents will scan the city to gather information.

References

1. Russel S. and Norvnig P.: Artificial Intelligence: A Modern Approach. Prentice Hall Series in Artificial Intelligence.
2. Bowling M.: RoboCup Rescue: Agent Development Kit. CS departement of CMU., Pitsburg. December 2000.
3. Menhaj M.B.:Computational Intelligence. Vol 1. AUT press.