Using Earliest Deadline First Algorithms for Coalition Formation in Dynamic Time-critical Environment

Omid Amir Ghiasvand and Maziar Ahmad Sharbafi

***Abstract*—Multi agent decision making is a challenging problem in Artificial Intelligence field of research. Coordination between different agents which may be homogeneous or heterogeneous has different complexities. Communication limitation, accessibility to local (non-complete) information, various world models for different agents are some of such problems. Coordination and cooperation have key roles in multi-agent decision making systems. Conclusively, coalition formation has been an attractive and practical approach in multi-agent coordination. An efficient algorithm is required to decide about optimum number of coalition and also number of agents in each coalition. In rescue environment a dynamic algorithm for coalition formation required as situation change. Rescue Simulation Environment is a heterogeneous multi-agent dynamic environment that its aim is to simulate large urban disasters and rescue agent activities and considered as an appropriate platform for associated research. In this paper we propose new algorithms based on Earliest Deadline First scheduling for assemble several rescue teams for various rescue missions. There is also presented team's disassemble after performance of rescue missions and new rescue team's assembly due to new situation in the environment. To address a real test case we choose RoboCup Rescue Simulation environment.**

***Index Terms*—coordination; multi-aegnt system; coalition formation; earliest deadline first; RoboCup; Rescue Simulation.**

I. Introduction

Decision making in complex and dynamic multi-agent system is one of the challenging area in Artificial Intelligence field [1]. In designing multi-agent systems, coordination is a central issue [2]. There are considerable efforts to achieve coordination, using distributed and centralized approaches. Researcher use different solutions like, Fuzzy decision making, learning, optimization techniques based on evolutionary algorithms and also methods inspired from nature in various problems [2]. The important point is that each solution just work for a given problem and it's really difficult to compare them with each other. In Other words, there is no general solution for coordination problem.

We have chosen the rescue simulation environment as a test bed for analyzing the performance of proposed solution, for its ability to evaluate the methods in difficult situations. Rescue simulation is a complex and dynamic multi-agent system that developed in a universal project under supervision of RoboCup project. Its aim is to simulate the circumstances of a city after an earthquake.

Earthquake is one of the most unpredictable natural disasters. It can cause buildings and bridges to be collapsed; bury civilians in ruined buildings and also can start fires. While we cannot predict the earthquake, we should be ready to manage the consequences of earthquake and to minimize the damages. Simulating an earthquake is one of the best methods to obtain such a preparation, if the simulation includes all human activities that simulate all aspects of the disaster like collapsed buildings, buried civilians and spreading fires. In addition to it, the extended version of such a system can be used as a decision support system during the real disaster and can play an advisor role for the human operators. For example it can suggest proper strategy to human rescue team according to current situation and previous experiences.

Rescue Simulation project has started at 1999 as one of the RoboCup International Competition League. In current state, the rescue simulation project is an excellent test bed for testing coordination and cooperation mechanisms. Each year in rescue simulation league several research teams participate and new methods are developed and evaluated during the competitions [3].

In this paper we are going to use coalition formation based on earliest deadline first algorithms for scheduling rescue activities. Earliest Deadline first (EDF) is a real time scheduling algorithms for dynamic scheduling of tasks. In this algorithm there is a queue of resource waiting for tasks. The resource manager assigns requested resource to a task based on remained time to perform it. It always selects the task with the earliest deadline. The assigned resources to a single task are called a coalition. The coalition sizes depend on the task size and its deadline. In section 2 we introduce rescue simulation environment. In section 3 we introduce Ambulance Team agent in more detail and explain about the decision making challenges to achieve a coordinated behavior. In section 4 and 5 we introduce the proposed method for ambulance team agent decision making.

II. Defenition of the Simulation Environment

In rescue simulation environment, rescue agents such as Ambulance team, Police force and Fire brigade agents work to reduce the damages to the city and people. After the occurrence an earthquake several buildings are ruined; people buried inside the collapsed buildings become numerous and some buildings ignite. If the fire brigades do not react in time, fires spread out and become uncontrollable. In addition, blocked roads and collapsed bridges limit the accessibility to the fire sites and victims that make the situation even worst. All rescue agents need to coordinate their activities to achieve their goal in controlling the

Manuscript received May 9, 2011.

This work was supported in part by the Mechatronics Research Laboratory, Electrical and Computer Engineering Department, Azad University of Qazvin, Qazvin, Iran.

O. Amirghiasvand is with the ECE Department of Azad University of Qazvin, Barajin Ave, Qavin, Iran. (e-mail: o.amirg@gmail.com).

M. A. Sharbafi is with the ECE Department of Azad University of Qazvin, Barajin Ave, Qavin, Iran. (e-mail: Sharbafi@qiau.ac.ir).

disaster.

In this section some explanations about the rescue simulation environment are explained in two parts. First, the structure of this simulation environment is described and next, some details about the Ambulance agents which is the subject of our test-bed.

A. ResCue Simulation Sructure

RoboCup Rescue Simulation System (RCRSS) include some independent modules that moderate by a central module called simulation kernel. Agent controllers are different RCRSS modules that control rescuer entity in the rescue simulation environment and communicate with the kernel module using network connections. For simplicity we just call them agents. In fig. 1 the general structure of RCRSS is displayed.

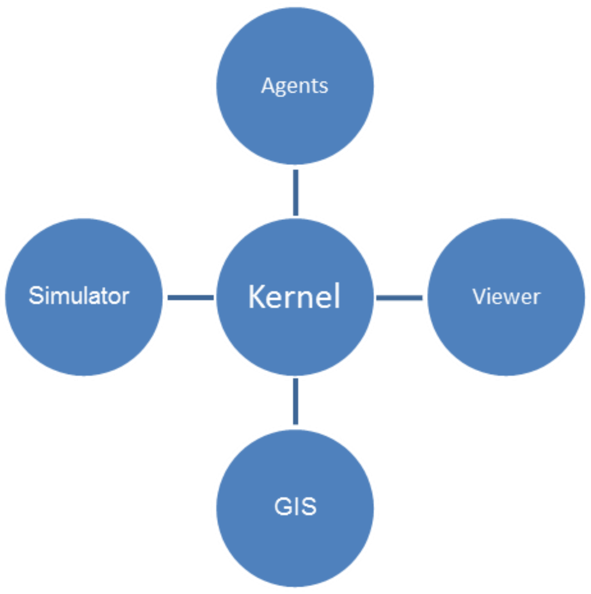

Fig.1.RCRSS Structure

To understand the RCRSS system in a sufficient depth, we need to explain some features of the system. There is much more detailed information in RCRSS manual [10].

Simulation period: The simulation period is 300 cycles and each cycle is equal to 1 min in real world. At the start of each cycle, the simulation kernel sends the sensory information of each agent to its controller module (after this, from our point of view the agent and the controller module are just the same). This sensory information includes the information of all objects in the radius of 10m from the agents, including some noises. The agent decides for his next action based on the sensory information and the current world model state, and sends his decision to the Kernel before the end of the cycle.

- Rescue Agents: There are 3 types of rescue agents which are Ambulance, Police and Fire brigade agents. Normally, there is about 15-20 agent from each type in a simulation scenario. Additionally, there is a center agent for each type to make communication between each other. So, three centers are Ambulance center, Police center and Fire brigade center. These agents have all computing abilities of other agents but cannot act directly in the environment, in other word they can think and decide like other agent but they are stationary and cannot act in the environment.
- Telecommunication: each rescue agent can communicate with other agents using channel based telecommunication system. In this way each agent can send and receive 4 messages in each cycle with the size of 4 bytes for each one. Each of center agents can send and receive twice the size of the number of its agents. Thus, according to this extra bandwidth, center agent can play a coordinator role.
- Agent action: Every agent (except the center agent) can send a command to the center agent, in every simulation cycle. There are two types of commands, which are general and especial commands. General commands are common between all agents like move and rest. There are some especial commands like rescue, clear, extinguish, load and unload that agents can send based on their types. For example ambulance agent can send a rescue, load and unload command to the kernel and cannot send extinguish.
- Scoring System: At the end of the simulation based on the number of saved civilians, and unburned buildings are simulation score is calculated as follow:

$$V = \left(P + \frac{S}{S_{init}}\right) * \sqrt{\frac{B}{B_{init}}} \tag{1}$$

In this equation, P is the total number of saved civilians, S_{init} and S are the total numbers of all civilians' health point at the start and the end of the simulation, respectively. B_{init} is the total number of building area, and B is the total number of unburned building area at the end of simulation.

B. Ambulance Team Agent

In this section, because of the great importance of saving civilians' life (see eq.(1)), the role of ambulance team agent in this cooperative work and especially the importance of coordination among them, a detailed description of this agent operation is presented. Ignorance about the roads' state (being blocked or not), balancing between search and rescue operation, time estimation for traveling in disaster area and finally hard condition of unreachable victim, add noticeable uncertainties that make decision making in coherent manner much more complicated .

Ambulance main duty is finding victims and carrying them to refuges. As aforementioned, each agent can just send one command to the kernel in each cycle. Mostly, the victims regarding to the volume of the debris (buried-ness), need more than one rescue action. Thus, the ambulance team agent have to spend several cycles (depends on buried-ness) to rescue each victim. For example, if the buried-ness is 30, the ambulance team agent should send 30 rescue commands in 30 cycles (each agent can send only one command in each cycle). If we add the load and unload actions, an ambulance need at least 32 cycles for rescue operation. There is also a travel time, that is the time required for traveling the path between victim positions to the nearest refuge. In addition to buried-ness, there are two other properties that affect the decision making of the ambulance team agents. The first one is health point (HP); which is normalized between 0 to

10000 for dead to completely healthy person. The second one is Damage which is the step of HP reduction in each cycle. Damage increases over time with a random rate, so it's very difficult to predict its value over time.

$$HP = HP_{old} - Damage \tag{2}$$

Because agent cannot predict the Damage, it's kind of impossible to predict when the civilian going to die (HP = 0). We use a statistical method to predict the time that the civilian going to die and call this value time to death property of victim. With assumption that our time to death property is acceptable and also other agent (Police and Fire brigade) help ambulance team by telling them about founded victim position and properties (HP, Damage and Buridness) Ambulance team agent or Ambulance center should decide about which civilian they should choose to rescue. In the following list there are some of the main parameters that influence the decision making process. Does not matter who is going to decide about choosing a victim (Ambulance Team or Ambulance Center):

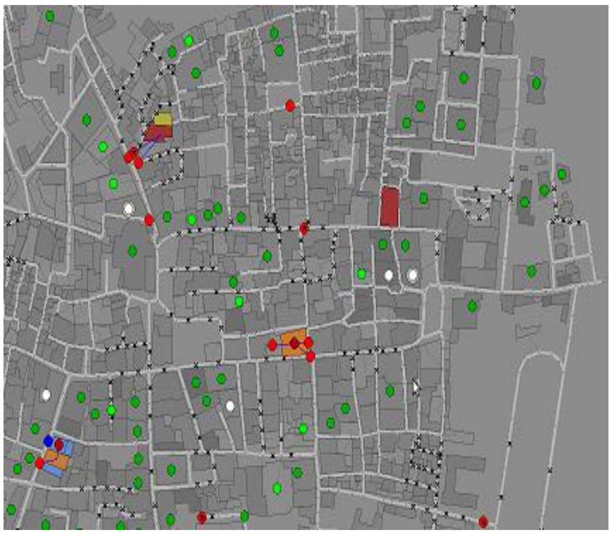

Fig.2.Two civilian near fire site at the top and oneblocked Fire Brigade at the bottom

- How many free ambulances (Ambulance that are not in the middle of rescuing a victim) can access the potential victim?
- Is it possible to rescue a victim before it time to death?
- Our information about the victims is still valid?
- Is the victim beside a fire site?

As you can see in figure 2, there are two civilian victims (green circle) beside the fire site (red building) and there is one civilian victim far away from fire site (safe place).

The main goal of ambulance team agent is rescuing maximum number of civilians. To achieve this goal ambulances need a sorted list of civilian victims to rescue. The list must be dynamic and update based on new received information. Actually the key point for maintaining such a list is the correctness of information inside the agent world model. No matter which methods we choose for synchronizing the world model of ambulance agents, there is always some inconsistency between agent world models because of communication system delay.

In our approach we choose earliest deadline first algorithms to sort civilian victims list. In the next section we are going to explain about our decision making approach from center based and distributed viewpoint.

III. Earliest Deadline First Algorithms

Earliest Deadline First algorithm (EDF) is one of the most applicable dynamic scheduling algorithms for scheduling real time system. Its places the process or task in priority queue. Whenever a scheduling event occurs (like task finishing) the queue will be search for process or task closest to its deadline. The task will be the next task to be scheduled for execution. In real time system there is deadline for each task and in order to achieve system objective the scheduler must try to meet all deadline. There is two type of real time system: hard real time and soft real time system. In hard real-time system if the task did not finish by the deadline it's totally useless. For example in rescue operation what happen if we rescued a victim after his or her death? So the task will be sorted based on remaining time to task deadline. And always the task with earliest deadline will be selected as a next task. In our case there is a good possibility that the next task (earliest deadline task) will be impossible to be done before the deadline and the victim gone be dead before the rescue operation finished. So we come to the conclusion that in this case it's much wiser to ignore the impossible mission and select the next possible task (rescue a victim that is possible to rescue in time). From this point of view and with such assumption (ignore impossible to rescue victim) the rescue environment is not a completely hard real time system. Of course we never can compensate a civilian death but when we ignore an impossible to rescue victim the rescue system still works and rescuer can continue their job and rescue other victim. In some hard real time system it's possible that the system completely shut down when the system fail to meet a deadline. Due to the specification of the earliest deadline first algorithms and its property, it's seems the algorithms is a good choice and have the proper feature and we can use this it for decision making in rescue environment and utilization of the rescue team will be maximized. In the next section we will explain about our center based decision making approach using EDF algorithms to form a temporary coalition of ambulances that cooperate to rescue the assigned target in time.

IV. Center Based Decision making Approach

As we explain in previous section, having an updated version of world model is crucial for optimum decision making. From the other respective, agents should have a consistent world model in order to be able to cooperate with each other. According to specification of the rescue simulation and communization delay it's very difficult to maintain a consistent version of world model for all agents. The following example will try to clear the significance of decision making for ambulance team agents. All ambulances can rescue a victim completely independent from the number of ambulances that is necessary for rescuing the victim on time. In this way we waste a lot of priceless ambulances time and instead of rescuing other victims we rescue a victim much more sooner than its deadline (time to death). For that reason if we could find a way for rescuing a victim just before its death time with exact number of ambulances that is necessary, we will make a breakthrough in rescuing civilian victim. To do so, there is need for an integrated decision making system that received information from all type of agents and after integrating information, select a victim as a target and then form a coalition to rescue the selected target. It seems pretty much clear that the best agent for playing such a role is ambulance center according to his communication bandwidth. In the next section we are going to explain using earliest deadline first algorithms along with coalition formation approach to coordinate ambulance agent's activity.

V. AMBULABCE PROBLEM

In order to use earliest deadline first algorithms for solving ambulance problem and form the coalition, first of all we need to map phrases in EDF algorithm to their equivalent in ambulance problem. In ambulance problem rescuing victims are the tasks and task deadline are the time to death for victims and finally selecting task is the processing in which a victim is selected by the ambulance center.

Ambulance center task selection, coalition formation and assigning the task to formed coalition are briefly describe as bellow:

1) If there is at least one free ambulance (unallocated ambulance) goes to step 2. In other case, wait.
2) Check the last selected victim. If it's need more ambulance assigns the ambulance to victim. If there is still more free ambulances go to step 3. Else
3) Approximate time to death for each civilian victim based on the last received civilian status information.
4) Sorting civilian victim based on the time to death.
5) Selecting first civilian victim as a candidate for rescue operation.
6) Calculating the number of ambulances needed to rescue the candidate in time. In other word the coalition size.
7) Check if it's possible to rescue candidate according to time to death? If not possible, remove the candidate from victims list go to step 3.
8) Form the coalition and assign ambulances inside the coalition to selected victim.

This above process execute each time one or several ambulance done their task and get free (they are added to unallocated ambulance list). You can find the pseudo code for ambulance center decision making in figure 3.

```
public void act()
{
  updateVictimListInformation();
  while (!availableAmbulances.isEmpty())
  {
    if (lastCoalition.needMoreAT())
    {
      addATToCoalition(lastCoalition)
    }
    else
    {
      for (Coalition coalition : coalitionSet)
        {
          if (coalition.needMoreAT())
            addATToCoalition(coalition)
        }
      if (availableAmbulances.isEmpty())
        break;
      sortVictimList();
      for (Victim victim : victimList )
        {
          if (victim.isRescueable())
            selectedTarget = victim;
```

Fig. 3. Ambulance Center decision making

To fully understand the algorithm, there is several terms and phrase we should explain:

- Need ambulances: Number of ambulances that the victim needed to rescued in time.
- Free ambulance: the ambulance that is not assigns to victims and waiting for target. Generally ambulances are our source.
- In time rescue: Rescue civilian victim before its death. We can count on both free and assign ambulances to determine if we could rescue victim in time. The pseudo code for determining if we could rescue a victim in time is presented in figure 4.

```
boolean  isRescueable(Victim v)
{
Int tw = 0; // total Work
for (Ambulance at : ambulanceSet )
{
  If ( at.timeToFree()+2 <
v.timeToDeath())
   tw += (v.timeToDeath()
at.timeToGet());
 }

  if (tw > v.buridness()+4)
     return true;
  else
     Return false;
}
```

Fig. 4. pseudo code for is rescue able function

is job and become free. This time depend on number of assign ambulances to ambulance target and the target buridness. For free ambulance this time is zero. In this way we can count on the works (rescue) of assigned ambulances in addition to free ambulances.

$$tw = \sum_{ambulance} (ttf - ttd) \qquad (3)$$

If the total work (tw) was bigger than remaining to death for the civilian we can save the victim and the ambulance center agent can assign ambulances to rescue the victim. We assume some extra work for some unforeseen event like blocked road when ambulance traveling to the refuge.

VI. Simulation Result

In order to test the proposed approach we select 5 scenarios used in RoboCup competitions on Kobe map. We test two version of code, in one version we use a fully distributed algorithms for ambulance decision making base on learning [3] and in the second version we use new approach for ambulance team decision making. There were not any changes in other rescue agent code. The result was shown in table 1. Because there is a small deviation of score even in two consecutive run of the same code we use an average score of three consecutive run as a final score for a scenario. The scores is shown in table 1

TABLE 1. Simulation Result

Map	New Ambulance	Old Ambulance
Kobe 1	88.57	62.91
Kobe 2	80.64	72.10
Kobe 3	74.67	34.69
Kobe 4	98.89	61.87
Kobe 5	93.45	51.23

In table 2 some of the result we got in RoboCup 2006 completions in comparison to other participating team is shown.

TABLE 2. RoboCup 2006 Result

	Foligno 2	Random 8	Kobe 1	Foligno 1	Random2
MRL	80.234	95.678	107.8	101.3	33.32
S.O.S	72.746	94.694	105.2	93.23	25.99
Impossible	58.337	88.621	106.4	63.03	22.72
Poseidon	68.560	95.08	104.1	86.64	0
IUST	68.747	93.204	104.5	97.19	25.37
SBCe	73.033	92.001	107.4	85.25	28.26

In figure 5 and 6 two screenshot from the final result of simulation are shown. The number of death civilian dramatically decreases in fig 6.

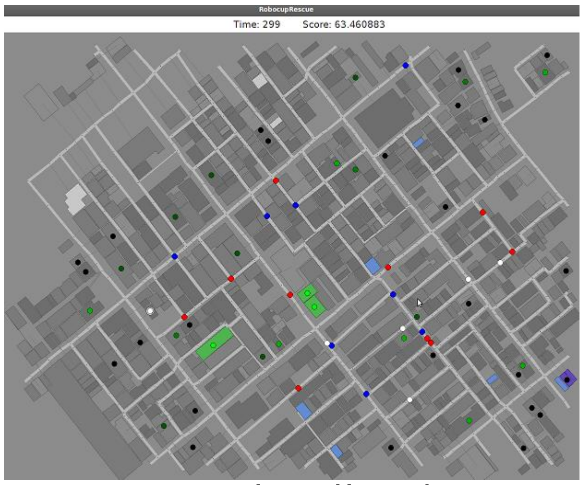

Fig.5. Result using old approach

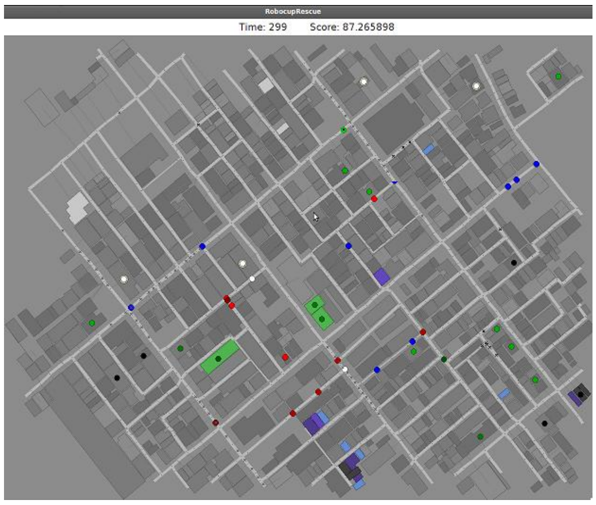

Fig. 6. Result using new approach

VII. Conclusion and Future Works

In this paper we use Earliest Deadline First algorithm to form coalition to rescue civilian victims. Using coalition formation mechanism to coordinate ambulance team agent behavior lead to an acceptable result as shown in simulation result section. Participating in RoboCup competition help us to compare proposed methods to other methods used by other research team working on rescue simulation problem. The MRL team becomes world champion in RoboCup 2006 and 2007. As a future work we are going to test noon preemptive EDF to improve the result by letting the coalition give up rescue operation in the middle of the process because of new funded victim that need immediate response.

References

[1] Ren, W., Beard, R., "*A Survey of Consensus Problems in Multi-agent coordination*", American Control Conference, 2005.

[2] Bedrouni, A.,Mittu, R., "Distributed Intelligence Systems", 2009, X, 182 p. 20 illus., Hardcover.

[3] Ahmad Sharbafi, M., Using Reinforcement-Emotional Learning in multidimensional and time-critical environment (Like Rescue Environment), M.Sc. Thesis, University of Tehran, 2006.

[4] Ahmad Sharbafi, M., Lucas, C., Amirghiasvand, O., Haghighat, A. *Using Emotional Learning in Rescue Simulation Environment*, Transactions on Engineering, Computing and Technology, 13, 333-337, 2006.

[5] Fazlollahi, B., Vahidow, R., "*Multi-agent distributed intelligent system based on fuzzy decision making*", International Journal of Intelligent systems, Vol.15, No.9, 2000.

[6] Chalkiadakis, G., Boutilier,C., "Coordination in Multi-agent Reinforcement Learning ", AAMAS'03, 2003

[7] Yan Meng Kazeem, O. Muller, J. C. "A Swarm Intelligence Based Coordination Algorithms for Distributed Multi-agent Systems", KIMAS, 2007.

[8] http://www.robocuprescue.org/wiki/images/Rules

[9] Tanenbaum, A. S.: Modern Operating Systems, Prantice Hall, 2009.

[10] http://www.robocuprescue.org/results.html

Omid AmirGhiasvand was born in Qazvin, Iran in 1982. He received his B.S and M.Sc. in Software Engineering from Azad University of Qazvin, Iran, in 2007 and 2010 respectively.
Since October 2004, Mr. AmirGhiasvand has been with the Mechatronics Research Laboratory (MRL), Azad University of Qazvin. He is the deputy director of MRL from 2007 and received different awards in RoboCup World cup competitions with. His research interest is multi-agent decision making system, software engineering for mobile robot and also operating system.

Maziar Ahmad Sharbafi is a Ph.D. student in University of Tehran from 2007. He received his B.S in Electrical Engineering/Control from Sharif University of Technology in 2003 and his M.Sc. from University of Tehran in the same major in 2006.
Since October 2004, he has been with the Mechatronics Research Laboratory (MRL), Azad University of Qazvin, Qazvin Iran. Mr. Sharbafi received different awards in RoboCup World cup competitions with MRL team during these years.
His current research interests include mobile robot locomotion control, multi agent decision making and bipedal robot motion control