RoboCup 2013 – Rescue Simulation League Team Description <Shabestar Rescue (Iran)>

Farshid Faraji, Shahin Akbarpour, Rahmat Hesami, Saeed Seyed Kazemi, Hekmat hajizadeh Asl

AI & Robotics Laboratory, Computer Group, Azad University of Shabestar
Iran

ffaraji@bonabiau.ac.ir, Akbarpour_shahin@yahoo.com,
saeed.ucna@gmail.com, hajizadeh@bonabiau.ac.ir

Abstract. One of the most important issues in RoboCup rescue simulation is Firebrigade agents' Decision making which have significant role in preventing fire spread through the city and saving civilians. FireBrigade agents to achieve a good performance need an efficient strategy to prioritize their goals. To improving Firebrigade agents' decision making an approach based on Learning Automata is presented.

1 Introduction.

RoboCup Rescue Simulation is an excellent multi-agent domain which includes heterogeneous Agents that try to cooperate in order to minimize damage in a disaster. These Agents must have effective cooperative behavior despite incomplete information. In fact, the main goal in this domain is minimizing the damage by helping trapped agents, extinguishing fiery collapsed buildings, and rescuing damaged civilians.

In a highly dynamic disaster situation as in RoboCup Rescue domain, it is a crucial capability of rescue teams to predict future states of the environment as precisely as possible. One of the most important issues in such a domain is to predict how disasters evolve over time, e.g. how the fire spreads in the city.

Firebrigade agents have important role in preventing fire spreading and saving civilians. While there are fiery buildings in the city, main role of Firebrigade agents is extinguishing them. In order to efficient deal with fires, Firebrigades should make some different groups. In most cases the number of fiery buildings is more than Firebrigades. On the other hand this problem becomes more acute because of their tank capacity. Each Firebrigade should spent significant time to fill its water tank. So estimating the number of firefighters to extinguish a fiery building and prioritizing their

tasks are major issues. It is difficult to do this estimation via normal techniques and needs complicated calculations. So it should be simplified and intelligent methods should be used.

Firebrigade agents' decisions are depend on many environmental parameters like buildings fieryness, area, temperature, type of building and so on. Thus, the agents should set a priority to these parameters to have more effect in the environment. Considering one parameter more than others by an agent cause the other parameters will be ineffective in decision making process and vice versa.

Determining a coefficient for mentioned parameters such that agents could make best decisions in all conditions is difficult. Many solutions and methods have proposed to these kinds of problems and in fact they work well in specific conditions. These methods can be divided in two general categories. The first category is heuristic and Trial and Error methods. These methods do not work well in all conditions. The second category of methods to determining the coefficients are methods based on artificial intelligence. These methods generally use techniques like Genetic Algorithms, Reinforcement Learning or Fuzzy logic [1][2][3][4].

These mentioned methods have two main problems. First, the rescue simulation environment is very large and those methods will work fine in the areas that can get responses rapidly. All mentioned methods need lots of runs and due to many scenarios in rescue simulation; the number of runs would also be much more. Second problem is mentioned algorithms have complicated structure and are slow and need very accurate evaluation functions also. In fact there is not any evaluation function to return accurate evaluation of parameters' coefficients. Therefore the methods that depend on to these evaluation functions like Genetic Algorithms, Fuzzy Logic and or algorithms based on Neural Networks could not be more efficient.

In this paper a method based on Learning Automata is proposed to overcome these problems and improve firefighter's decision making. For this purpose a Learning Automata with fixed structure to prioritize Firebrigade agents' goals is used.

2 Performance of Firebrigade Agents.

There are lots of parameters that affect Firebrigade Agents performance. In previous section we have mentioned some kind of these parameters that are main reason of fire spread in the environment. But the main question is that whether considering these parameters are sufficient to Firebrigade agents' success?

Agents that only rely on some general parameters like mentioned in above will not be successful in many scenarios. The main reason of failures in such conditions is that performance of rescue teams is depending on the number of survived civilians. Although further expansion of fires affect the performance of rescue teams, but surviving civilians are more valuable.

In order to dial with these kinds of problems we have considered the following parameters in our Firebrigade agents' decisions.

- Building Fieryness
- Building Temperature
- Building Area
- Distance form civilians
- Distance from safe area
- Distance from fiery area
- Building type
- Simulation time

Our Firebrigade agents' general trend is that assigns a priority (coefficient) for each parameter and then by combining these priorities in somehow, determines a priority for each building. After doing this for all buildings, agents can choose the buildings that have the most priority to extinguish. The priorities should be determined in a way that all parameters affect the environment [5]. We have used Learning Automata to determine these priorities.

3 Learning Automata

As a model for learning, Learning Automata act in a stochastic environment and are able to update their action probabilities considering the inputs from their environment, so optimizing their functionality as a result.

A variable structure automaton is represented by a sextuple ⟨ α, β, Φ, P, G, T⟩, where β is a set of inputs, Φ is a set of internal states, α is a set of outputs, P denotes the state probability vector governing the choice of the state at each stage k, G is the output mapping, and T is the learning algorithm. The learning algorithm is a recurrence relation and is used to modify the state probability vector. It is evident that the crucial factor affecting the performance of the variable structure Learning Automata is the learning algorithm for updating the action probabilities. In linear reward penalty algorithm (Lrp) scheme, the recurrence equation for updating P is defined as:

$$P_j(k) = \begin{cases} P_j(k) + a\left(1 - P_j(k)\right) & if\ i = j \\ P_j(k)(1-a) & if\ i \neq j \end{cases} \quad (1)$$

if β(k) = 0 (i.e. reward received from environment) and

$$P_j(k) = \begin{cases} P_j(k)(1-b) & if\ i = j \\ \dfrac{b}{r-1} + (1-b)P_j(k) & if\ i \neq j \end{cases} \quad (2)$$

if β(k) = 1 (i.e. penalty received from environment).

In writing the above formulations, we have assumed that the automata had chosen the action with the subscript I at stage k-1, so it obviously should update its action probabilities depending on its environment's response received at stage k. The parameters *a* and *b* represent reward and penalty parameters, respectively. The parameter a(b) determines the amount of increase (decreases) of the action probabilities [6][7].

4 Our proposed model based on Learning Automata

Fire brigade agents to prioritize their targets use a Learning Automata system with 8 states. The states are as follow:

- State1: Increasing the priority of Building Fieryness.
- State2: Increasing the priority of Building Temperature
- State3: Increasing the priority of Building Area
- State4: Increasing the priority of Distance from Civilians
- State5: Increasing the priority of Distance from safe area
- State6: Increasing the priority of from fiery area
- State7: Increasing the priority of Building type
- State8: Increasing the priority of Simulation time

In fact the Learning Automata by deploying in each of the states increases the priority of the relevant parameter. The whole process of Learning Automata is that the system start up from state 1 and after increment of corresponding parameter, new test with new parameters is done. During the learning phase, the output of environment is evaluated as Learning Automata result in various states.

If the obtained result is better than the previous result then Learning Automata assumes that its action was effective and then repeats the procedure again. This process keeps the Learning Automata in current status and in the next times the priority of corresponding parameter increases. While the result of next state is better than the previous state, Learning Automata repeats the procedure and increases the priority of parameters.

In case of the result was worse than the previous result and the final score of agent decreased, the system assumes that the amount of current priority of parameter should not be increase by considering other priorities. At this time the system penalize the priority of that parameter and sets its value to half of current amount and the state of Learning Automata changes to next state. The probability of choosing next state is updating after each process. All states initially have probability value equal to 0.1 to be selected.

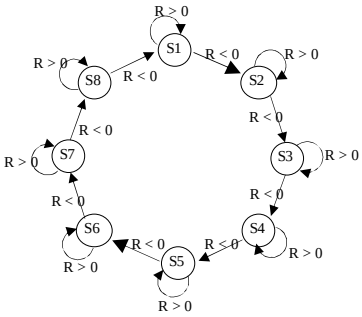

Fig. 1. Learning Automata structure to adjust priorities

An important point in choosing next state is that if current state of system gets a penalty, that state will not be selected again as next state. This condition makes the system to select various states so the system will converge to the optimal priorities.

Priority of parameters is changed when they gets reward or penalty. With each reward the value of priority becomes double. This process is repeated until the current state of Learning Automata system receives penalty. At the first penalty the value of priority is halved and then if that state becomes the current state in the next steps and receives reward, the amount of reward will be 1 at this time and the value of priority will increased 1 unit. This 1 unit reward will be continued until the value of priority becomes equal with the value in previous state which received penalty. If it happened and the state has got reward again, the amount of reward will be doubled again.

This process of reward and penalty system causes to get accurate values of priorities. Each penalty means that the answer is near optimal value or it had increased unbalanced. Thus its value is halved after penalizing. It means that possibly the answer is between current value of priority and its double. So by each reward the value of priority is increased one unit.

Decision making algorithm for Firebrigade agents work as follow: the priorities of mentioned parameters which calculated by Learning Automata system for each building is multiplied to the value of its related parameter and the result is used by agents with some changes. In order to prevent sudden increase or decrease of result, it is normalized between minimum and maximum range like fig 2.

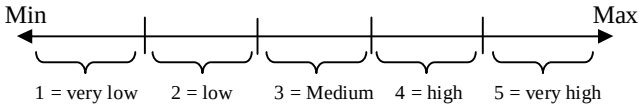

Fig. 2. Defined range for a building among other buildings

The number of ranges is obtained experimentally. As regards the parameters which Fire brigade agents use in their decision algorithm, following results are obtained:

4.1 Establishing coordinated groups using Learning Automata

In learning phase each agent selects a nearest building or a building in fiery area to extinguish using mentioned parameters. By accurate determination of conditions related to these parameters, at the beginning of simulation, the Firebrigade agents that are near a specific fiery building work together and extinguish it. By proceeding simulation time agents make dynamic groups and extinguish fires in deferent parts of city.

4.2 Choosing suitable fiery building for extinguish

The overall priorities after learning phase leads to determine a suitable fiery building for extinguish. In fact the main goal of Learning Automata system here is selecting optimal and suitable fiery buildings.

4.3 Determining fire spread direction

Fire spread direction and its speed in rescue simulation environment is depending on type, fieriness and temperature of buildings. These parameters are considered in Learning Automata system and an accurate priority is calculated for them. So agents determine speed and direction of fires in city and by extinguishing suitable fires can control fires.

5 Police Office and Police Forces

The responsibility of police force is clearing the blocked roads of the city. One of the most important things which have a great impact on agents' prosperity in rescue operation is exploration and rummaging disaster environment as soon as possible [8]. By beginning disaster phenomenon, the disaster environment should be explored and blocked roads, fiery buildings, and buried civilians should be found. The main part of exploration operation is police agents' responsibility.

We are using an approach based on ant colony optimization algorithms [9] and reinforcement learning algorithms that improves the power of research and exploration effectively. By using of this way, our agents can explore more areas in minimum time [8]. This approach by adjusting ant colony optimization algorithm parameters via Learning Automata, give a powerful ability to agents to search and explore environment.

Each agent uses the ant colony optimization mechanism to select next node as follows:

$$p_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha} . [\eta_{ij}(t)]^{\beta}}{\sum_{j_k=1}^{n} [\tau_{ij_k}(t)]^{\alpha} . [\eta_{ij_k}(t)]^{\beta}} \quad (3)$$

In the above formula p_{ij} is the probability of selecting edge between node *i* and node j as path, $\eta_{ij} = 1 / d_{ij}$ where d_{ij} is the distance between node i and node j, and τ_{ij} is the intensity trail on edge (i, j). Parameter α referred to the important of pheromone effect and β referred to the important of distance between node i and node j [9].

In addition to use ant colony algorithm for search and exploration, Police agents are equipped to a Learning Automata which adjusts the parameters of ant colony algorithm online. The most important parameter of this algorithm which can balance exploration and exploitation is β parameter that expresses the importance of edge cost. To adjust this parameter we are using a fixed structure Learning Automata. Fig. 3 shows the structure of this Learning Automata.

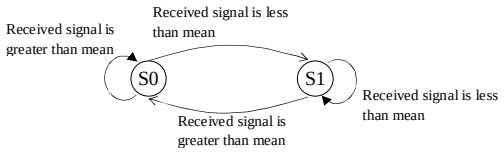

Fig. 3. Learning Automata structure for adjusting parameter β

The received signal is the period of time takes to get from origin to destination. The mean amount is getting by shortest path algorithm.

If the automaton is in state S0 and the received signal is less than mean amount then automaton keeps it current state and the amount of β parameter is reduced about differences between received signal and mean value. This process leads to have more exploration in the future. If the automaton changes its state from S0 to S1 it means that we will have more exploitation in the future because the amount of β will be increased.

One of the advantages of this approach is that in the first cycles of the simulation it mostly uses the shortest path between source and destination and gradually other paths are used and this leads to more areas being explored through moving between points. The peak point of exploration is at the beginning of simulation; hence this is one of the most important characteristics of this approach.

6 References

1. S. Luke and L. Spector, Evolving teamwork and coordination with genetic programming," in Proceedings of the First Annual Conference on Genetic Programming. MIT Press, pp.150-156.1996.
2. J. K. Raphael Crawford-Marks, Lee Spector, Virtual witches and warlocks: A quidditch simu-lator and quidditch-playing teams coevolved via genetic programming," in Late Breaking Papers of the Genetic and Evolutionary Computation Conference (GECCO-2004), 2004.
3. Aghazadeh O., Sharbafi M. A. and Haghighat A.T, Implementing Parametric Reinforcement Learning in Robocup Rescue Simulation, Lecture Notes in Computer Science, 2008, Volume 5001/2008,pp. 409-416.
4. Radmand, A., Nazemi, E., Goodarzi, M.: Integrated Genetic Algorithmic and Fuzzy Logic Approach for Decision Making of Police Force Agents in Rescue Simulation Environment. In RoboCup, pp. 288-295, 2009.
5. Kitano H., Tadokoro S., Noda I., Takashi H., Shinjou A., Shirmada S.; "RoboCup Rescue: Search and Rescue in Large Scale Disasters as a Domain for Autonomous Agents Research", IEEE Conference on Man, Systems, and Cybernetics, 1999.
6. Narendra, K. S. and Thathachar, M. A. L., "Learning Automata: An Introduction", Prentice Hall, Inc., 1989**.**
7. [8] M. A. L. Thathachar and Sastry, "Varieties of Learning Automata: An Overview", IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, vol. 32, no. 6, 711-722, 2002.
8. Mohammad Reza Khojasteh, Farshid Faraji, Mostafa Asghari, Maghsoud Kafshnochi: RoboCupRescue 2008 – BonabRescue Rescue Simulation Team Description.
9. Macro Doring, " The Ant System : Optimization by a Colony of Cooperating Agents", IEEE Transactions on Evolutionary Computing, Vol 1, no.1, 1997, pp.53-66, JOURNAL.