A Distributed Cooperative Reinforcement Learning Method for Decision Making in Fire Brigade Teams

Abbas Abdolmaleki[1,2], Mostafa Movahedi, Nuno Lau[1,3], and Luís Paulo Reis[2,4]

[1] IEETA – Institute of Electronics and Telematics Engineering of Aveiro, Portugal
[2] LIACC – Artificial Intelligence and Computer Science Lab., Porto, Portugal
[3] UA – University of Aveiro, Campus Universitário de Santiago, 3810 193 Aveiro, Portugal
[4] EEUM - School of Engineering, University of Minho - DSI, Portugal
Campus de Azurém 4800-058 Guimarães, Portugal
{Abbas.a,nunolau}@ua.pt, mr.mos.movahedi@gmail.com,
lpreis@dsi.uminho.pt

Abstract. Decision making in complex, multi-agent and dynamic environments such as disaster spaces is a challenging problem in Artificial Intelligence. This research paper aims at developing distributed coordination and cooperation method based on reinforcement learning to enable team of homogeneous, autonomous fire fighter agents, with similar skills to accomplish complex task allocation, with emphasis on firefighting tasks in disaster space. The main contribution is applying reinforcement learning to solve the bottleneck caused by dynamicity and variety of conditions in such situations as well as improving the distributed coordination of fire fighter agent's to extinguish fires within a disaster zone. The proposed method increases the speed of learning; it has very low memory usage and has a good scalability and robustness in the case that the number of agents and complexity of task increases. The effectiveness of the proposed method is shown through simulation results.

Keywords: RoboCup Rescue Simulation, Multi agent system, Fire Brigade, Decision Making, Reinforcement Learning.

1 Introduction

Several authors have proposed general models for flexible coordination of agents. However, most of the approaches either are not sufficiently reactive to perform efficiently in real time and dynamic domains or do not provide agents with sufficiently developed social behavior to perform intelligently as members of a team in continuous, multi-objective and complex multi-agent environments [1]. Notable research may be recognized in Stone's and Veloso's work [2] that has been applied with success to RoboCup soccer and network routing.

In order to achieve complex behavior acquisition using machine learning methods, Stone and Veloso [3] proposed to introduce a layered learning system with basic skills such as "shootGoal", "shootAway", "dribbleBall", and so on. Kleiner et al [4] also proposed a multi-layered learning system for behavior acquisition of a soccer robot.

X. Chen et al. (Eds.): RoboCup 2012, LNAI 7500, pp. 237–248, 2013.

Most implementations of multi-agent coordination frameworks rely on domain specific coordination. Some relevant exceptions may be identified however. An example is Jennings' joint responsibility framework [5], which is based on a joint commitment to the team's joint goal and was implemented in the GRATE* system. One other approach to automated planning is discussed in hierarchical task networks (HTN) where the dependency among actions can be given in the form of networks [6]. Regarding one possible and commonly used test bed for the framework, the Robotic Soccer competition [7, 8, 9], the UVA Trilearn team's coordination graphs provide a way to parameterize a coordination structure over a continuous domain [10].

To manage coordination between agents three options are proposed in [11]: environment partitioning, centralized direct supervision and decentralized mutual adjustment. Among these three approaches, the decentralized approach is more flexible but this does not necessarily provide better results. Considering Robocup rescue simulation as another test bed, in [12] a hybrid approach is proposed to take advantage of both centralized and decentralized approaches. In order to make a decision about the number of ambulances which should cooperate in rescue civilian, evolutionary reinforcement learning is utilized by [13]. In [14] a fire brigade learns to do task allocation and how to choose the best building to extinguish to maximize the final score. In [15] fire brigades learn how to distribute themselves within the city using neural reinforcement learning.

The authors recently developed a reinforcement learning based model for optimizing the task of a single fire fighter agent [16]. This model uses a reinforcement learning method based on the Temporal Difference (TD) methodology. Temporal difference methods update the estimated data without waiting for the final outcome, allowing prompt reactions in complex environments like disaster spaces. In this paper we have extended this approach and developed a distributed coordination and cooperation method based on reinforcement learning to enable a team of homogeneous, autonomous fire fighter agents, with similar skills to accomplish complex allocation of tasks, with emphasis on extinguishing fires task in disaster space. The major contribution made in this paper is to introduce a distributed cooperative algorithm based on reinforcement learning for firefighting agents to save a city and civilians from fire at a fire site[1]. It is a distributed coordination because each agent makes its decision by itself and there is no central command. The rest of this paper is organized as follow. In next section the problem is detailed and assumptions are presented. In Section 3 we explain the test bed which is used to implement and test our approach. Section 4 discusses how to design and model the problem for applying reinforcement learning, in detail. In Section 5 details of implementation are presented and achieved results are shown. Section 6 concludes.

2 Problem Formulation

Disaster space firefighting presents a complex task allocation problem in a very dynamic and uncertain environment. Fire brigades are responsible for controlling fire. The most important issue is to select and extinguish the best fire point to reduce damage of the civilians and city. We have decomposed the firefighting task (Fig. 1) into some important subtasks. The two most important subtasks are:

[1] A fire site is an area within a neighborhood consisting of burning buildings.

1- In which way to allocate large number of resources (Firefighters agents) to various fire sites.
2- How allocated agents cooperate and coordinate in a decentralized way at each site to control the fire.

In this paper we propose a solution for the second subtask which is seeking an acceptable distributed cooperation strategy for firefighter agents to control a fire site using reinforcement learning.

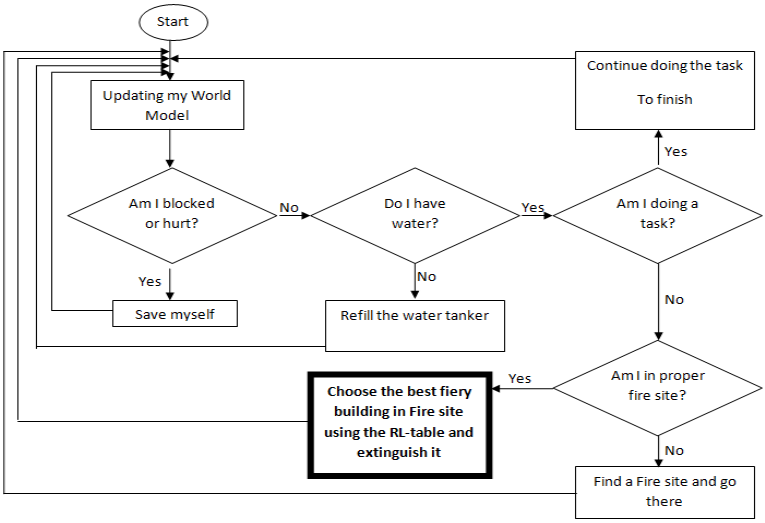

Fig. 1. Flowchart of firefighting task, the bold box is the sub-task which we have focused on in this paper

Like humans, robots potentially work better as a team than working alone in challenging domains. By cooperation, robots can complete the tasks faster, with more robustness and also complete tasks which are impossible for a single robot. However for effective cooperation, there are many difficulties in uncertain and dynamic environments, such as the presence of noise, communication problems, resource failures, and difficulties of mobility and computation.

Considering the aforementioned challenges we make some assumptions to present a distributed cooperative reinforcement learning method, which are as follows:

#Assumption 1: All the agents or resources have the same knowledge of the fire sites and fiery buildings. It means we assume that the communication between robots is good enough to share their knowledge about the environment.

#Assumption 2: No routes are blocked, so there is no problem of mobility for agents.

#Assumption3: As firefighter agents need to refill their water tanks, we assume there are stations from which firefighters can do so, so there is no resource failure.

In a rescue simulation, fire fighter agents must perform a sequence of actions to carry out their firefighting task at a fire site efficiently. Each action alters the environment and also influences their choice of the next action. The goal of the agents is to achieve the best score at the end of the simulation. Agents should select a sequence of actions

which maximizes the score. Due to the limits of cognition of the agents and the fact that agents don't know a priori the effect their action will have upon the environment, using reinforcement learning is a suitable method to solve this problem. The mechanism which we have used to teach the agents is based on the Temporal Difference method of reinforcement learning [17]. Since TD methods update their estimates based in part on other learned estimates, without waiting for a final outcome (bootstrapping), this method is applicable for complex environments such as rescue simulations in which an action should be selected in each cycle.

Unfortunately, finding optimal planning solution for Multi-agent systems challenges are typically NP-hard. The goal then, is to produce acceptable and efficient distributed cooperative planning solutions based on reinforcement learning.

Our algorithm is based on an offline look-ahead search to find the best action to execute at each possible situation. In this problem, first we should define proper states and actions so that the learning task is to find the best action for each situation. The output of the learning would be a table in which for each possible situation there is an associated action.

3 Test Bed

RoboCup rescue simulation is a simulation of environment of a city where an earth quake has happened. The aim of creating this environment is to learn the best rescue strategies for humans and also fire extinguishing tactics to be used within earthquakes in real life. It is currently a major league in RoboCup simulation competitions. The simulation consists of a kernel as the main part of the simulation, and some simulators which simulate the earthquake, fire,traffic, blockades and civilians of the city and also a viewer which shows how the city changes over time. There are three kinds of agent: Police Force, Ambulance and Fire Brigade Agents which are present in the simulation to perform the rescue operation. Police Force Agents control the traffic in the city and open blockades of closed roads. The Ambulance agent's duty is to locate the injured civilians and give first-aid to them until they are retrieved from collapsed buildings as well as carrying them to refuges. Fire simulator used to simulate the spread of fire over buildings, taking into account building material, wind and temperature of building. The main job of the Fire Brigade Agents is to extinguish fires by spraying water in its tanker (the capacity is limited) on buildings which decreases the temperature of buildings. If fire fighter agent sprays water on a cool building that building will get water damage. Traffic simulator determines how agents navigate around on roads and buildings. The performance score of a simulation is computed from the quantity of surviving civilians and the extent of damage to the city. When the fire brigade agents want to extinguish fire they have to prioritize which of the burning buildings to extinguish and in which order. We have applied our method to this environment to find an policy for finding an action in each environmental state.

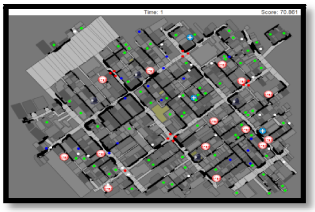

Fig. 2. Rescue Simulation Environment

Fig. 2 shows a screenshot of the simulation environment. It displays map of Kobe city after an earthquake. The blue, white and red circles represent the Police force agents, Ambulance agents and Fire brigade agents respectively. The bright and dull green circles display healthy and hurt civilians and black circle represent the dead civilian. The yellow, orange, dark red and black building represents level of fieriness of a building. Yellow shows the lowest level of fieriness of a building and black shows a burned out building. There is also a special type of buildings: the buildings that are marked with home icon are refuges where saved civilians are taken. Black areas on the roads represent blockades. The simulation runs for 300 time steps.

In order to reduce the complexity of simulation process we designed and implemented a simple rescue simulation environment according to assumption 1-3. This system has all features we need such as a fire and traffic simulator. Our simple rescue simulation is much faster than the official rescue simulation while maintaining the necessary capabilities. In Fig. 3 a screen shot of the simple rescue simulation is displayed. This environment has a fire simulator with the same algorithms of official Rescue Simulation since it is a standard system. Burning of buildings and fire spread is the same as the original Rescue Simulation. Also the algorithm to calculate the health of civilians is the same as the one in misc simulator[2] of Rescue Simulation.

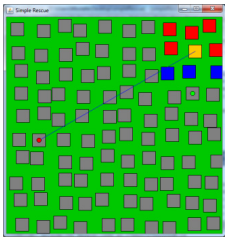

Fig. 3. Simple rescue simulation

The learning processes of agents and test phases are performed within this simulator and the results potentially can be used in original rescue simulation.

4 Design

This section presents necessary issues including a description of the environment, the design of the reward function and the learning algorithm used in our approach. Finally a proposed learning procedure for the problem is presented. But first a brief explanation of reinforcement learning is presented.

4.1 Description of Environment

In this section we define actions for agents and possible states for environment.

First the possible actions for agents should be determined. So agents in each state can perform one of following actions:

[2] The misc simulator calculates the health of civilians within Rescue Simulation.

1- extinguishEasyestBuilding: Extinguish the building that has lowest temperature in firesite.
2- extinguishNearestToCivilian: Extinguish the nearest fiery building to a civilian in firesite.
3- extinguishNearestToCenter: Extinguish the fiery building in a firesite which is closest to the center of the city.
4- extinguishNearestToMe: Extinguish the nearest building in a firesite to the firefighter.

These four actions are the most reasonable actions that a firefighter agent can take in different situations.

Although in Temporal Differential methods the model of the environment is not required, it needs a clear description of the states of the model. So in this section we describe the rescue simulation environment with some discrete and finite states. The states of environments are modeled by the following features.

1- freeEdges: The number of edges that fire can spread from represented by 0,1,2,3 or 4. For example in Fig.3 the fire can spread only from the left side, so the number of freeEdges is 1.
2- distanceFromCenter: The distance of nearest fiery building to the center of city. This is low, medium or high.
3- distanceFromCivilian: The distance from the nearest fiery building to a civilian. This is low, medium or high
4- volumeOfFieryBuildings: The total volume of fiery buildings, classified as either veryLow, low, medium, high, veryHigh or huge.
5- IsExtinguishEasyestBuilding: If currently any firefighter is extinguishing the easiest building (action 1) in firesite. This is True or False.
6- IsExtinguishNearestToCivilian: If currently any firefighter is extinguishing the nearest fiery building to a civilian (action 2) in the firesite. This is True or False.
7- IsExtinguishNearestToCenter: If currently any firefighter is extinguishing the nearest fiery building to the city center (action 2) in the firesite. This is True or False.
8- IsextinguishNearestToMe: If currently any firefighter is extinguishing nearest fiery building to itself (action 4). This is True or False.

The states 1 to 4 are enough to describe the environment for an agent. To allow agents to cooperate effectively they need to know about other agents. So states 5 to 8 describe the other agents in an environment. These states give an idea to agents what the other agents are doing at each step which provides them with a distributed cooperation. Good communication between agents is therefore required, and this is assumed in the problem formulation section.

There are $5\times3\times3\times7\times2\times2\times2\times2=4320$ different environmental states. These states can describe all situations in all fiery scenarios.

The problem is to find the best action in each state that leads to best score at the end of the simulation.

4.2 Reward Function

The reward function plays an important role in reinforcement learning as it directs the learning process towards a solution. In a burning city, the world will never improve

with time; hence the agent will always incur a penalty in the reward function. The fire brigade must minimise the penalty score.

The reward function used is defined as follows:

- For each *waterDamaged* building (a building which is extinguished and is damaged because of too much water) in each time step the agent receives a reward -1.
- For each fiery building in each time step the agent receives a reward -2.
- For each burned out building in each time step the agent receives a reward -3.
- For each damaged civilian in each time step the agent takes reward *HP-1000*. The *HP* (Health Points) of each healthy civilian is 1000 and if the building that contains a civilian ignited, the civilians' *HP* reduces over the time.

The first reward causes the fire brigade to extinguish fire without wasting water. The second and third rewards force the fire brigade to extinguish fire as quickly as possible and the final reward motivates the fire brigade to save civilians from fire.

4.3 Learning Algorithm

Fire brigade agents learn which of their possible actions are most valuable (leads to the best score) for a particular environmental state. To do this, the agents are taught by an on-policy TD control method called SARSA [17]. Reward values (Q values) of each action in each state were updated by formula (1) considering their rewards or penalties.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (1)$$

In above equation s_t is previous state before taking action a, s_{t+1} is current state after action a, r is reward received from the environment after taking action a in previous state and reaching the current state, a is the last executed action, Q is the Q-Table, *alpha* is learning rate and *gamma* is discount factor.

4.4 Learning Procedure

In reinforcement learning methods rewards provide guidance to the agent. If an agent receives high rewards it understands that it performed a useful action sequence; conversely poor rewards suggest a poor choice of actions. If an agent earns low rewards most of the time, convergence is poor and the best action cannot be found. In a very complicated environment such as rescue simulation, the problem search space is huge. In this case the agent may continually get bad rewards and so not learn the optimum policy. To overcome this problem we carried out our learning phase step by step. First we trained just a single agent in 24 scenarios ranging in difficulty from easy to complicated which is explained in [16]. Then we used the obtained Q-table (ignoring cooperation states which reference other agents) from the previous step as the initial knowledge of four agents. Next, we started training four agents in 8 different scenarios. In each scenario the firefighters randomly use one of the obtained Q-Tables from its previous learning scenarios as the initial knowledge to output an

improved Q-Table containing more records and which has been informed by a greater number of states.

5 Implementation and Results

This section presents the results of using the previously explained method in simplified rescue simulation. In this section the scalability and robustness of the method will be discussed. We will discuss the results in three subsections. First the learning procedure and results in training scenarios will be presented. Then the effectiveness of the method on three test scenarios is shown. Finally we will demonstrate the scalability of method by changing the number of agents.

5.1 Cooperative Multi-agent Learning

In order to train the firefighters, the parameters γ and α of the Q-Table are initially set to 0.5 and 0.7 respectively, based on trial and error. The value of e factor in the *e-greedy* selection algorithm is set in each episode based on the formula $e = e * 0.9^{episode}$. This formula causes agents in early episodes to favor exploration of the search space and in later episodes to use obtained experience from previous episodes to try to converge to a solution. First we trained just a single agent in scenarios ranging from easy to difficult [16]. Then, in order to teach coordination between agents, 8 scenarios, each using 4 agents were simulated. In the first scenario the initial Q-Table for all agents is the Q-Table obtained from the previous training of a single agent. In successive simulated scenarios the initial Q-Table is the same for all agents and is chosen randomly from the recorded Q-Tables of each agent at the end of the preceding scenario. The end condition of each training phase for a scenario is when agents converge to a policy which does not change for 20 episodes. This will be considered as a learned policy by agents. It was interesting to note, that at the end of all eight scenarios, all the agents had converged to produce the same Q-tables. Agents learn appropriate actions for each state after training in all scenarios, because the scenarios and learning procedure have been carefully designed so that every state is visited several times. In table 1, characteristics of training scenarios and the result of training is presented. Score in table 1, is calculated based on number of dead civilians and burnt out buildings.

Table 1. Main characteristics of the scenarios used for cooperative learning.and results

Scenarios	FireSite Position	Initialized amount of fire	Position of civilians related to Firesite	Convergence Episode	Score
Scenario1	City Center	Small	Surrounding	55	9.977 out of 10
Scenario2	City Center	Medium	Surrounding	101	8.795 out of 10
Scenario3	City Center	Medium	West	116	10.787 out of 11
Scenario4	City Center	Medium	East	128	8.611 out of 9
Scenario5	North West corner	Small	East and South	41	9.891 out of 10
Scenario6	North West corner	Medium	East and South	74	9.629 out of 10
Scenario7	North West corner	Big	South	73	8.973 out of 10
Scenario8	South	Big	Surrounding	59	11.31 out of 12

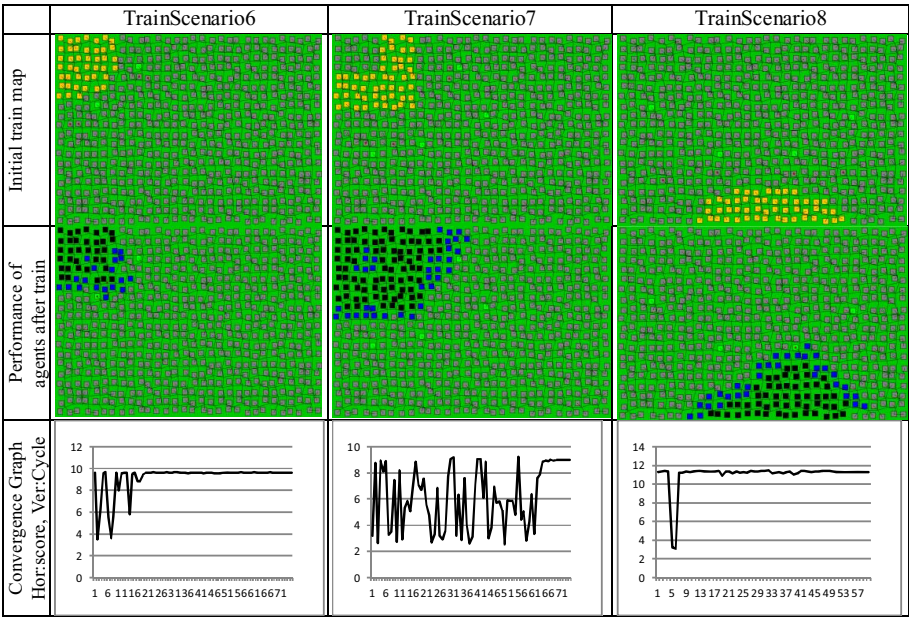

Fig. 4. Training scenarios(6,7,8) and performance of agents on train scenario after learning as well as convergence graphs

Fig.4 shows graphically Training scenarios and performance of agents on 3 train scenarios after learning as well as convergence graphs. The achieved result proved that agents after training can control fire effectively.

5.2 Evaluation in Test Scenarios

After training 4 agents in 8 different scenarios, we tested these 4 agents in 3 new scenarios and compared the performance of these agents with agents which were trained using a single agent approach [16]. To analyze the performance of trained agents, the test scenarios used were more difficult than the scenarios in which the agents were trained. Table 2 presents the test results and Figure 5 shows graphically the performance of agents in 2 train scenarios (1 and 2). The results show the cooperatively trained agents have much better performance than agents which were trained in non-cooperative mode. Also, the results achieved by cooperative agents are very good for complex scenarios, as the agents saved all civilians and controlled fire effectively, limiting fire damaged areas. This shows that the method provided a robust approach which can be applied in scenarios that are very different from those used in the training phase.

Table 2. Results in test scenarios for cooperative and non cooperative learning approaches

Scenarios	Time to complete task		Number of burned out buildings		Score		Max score
	Without Coord.	With Coord.	Without Coord.	With Coord.	Without Coord.	With Coord.	
Scenario 1	436	290	410	278	2.017	9.15	12
Scenario2	490	203	541	115	2.564	10.808	12
Scenario3	473	410	570	337	0.27	5.582	10

5.3 Evaluation of Scalability

To test the scalability of proposed method, we used scenarios where the number of firefighters is different than the number of agents used in the training scenarios. The results (presented in table 3) show that the agents can cooperate well and also that their performance is better than firefighters trained in non-cooperative mode.

Table 3. Results in test scenarios where the number of firefighters is different than the number of agents used in the training scenarios for cooperative and non cooperative learning approaches

Scenarios	Number of firefighters	Time to complete task		Number of burned out buildings		Score		Max score
		Without Coord.	With Coord.	Without Coord.	With Coord.	Without Coord.	With Coord.	
Scenario 1	7	361	246	268	167	8.221	10.257	12
Scenario2	8	292	113	160	77	10.34	11.198	12
Scenario3	8	456	294	318	112	6.76	8.788	10

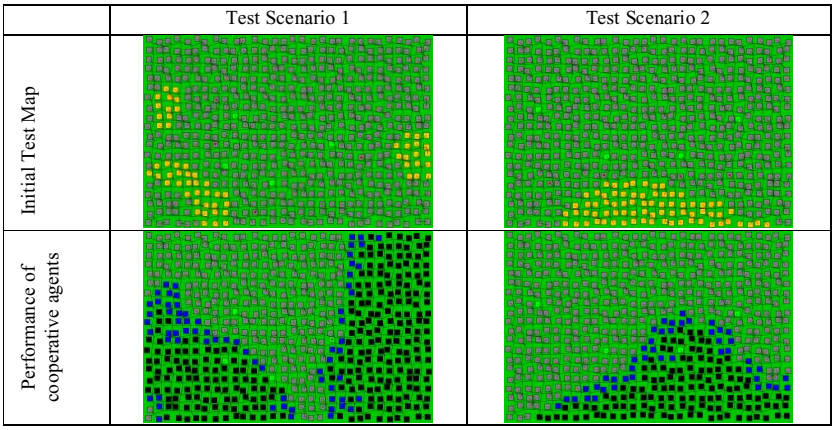

Fig. 5. Results in test scenarios, Comparison of cooperative agents and non-cooperative agents in test scenario

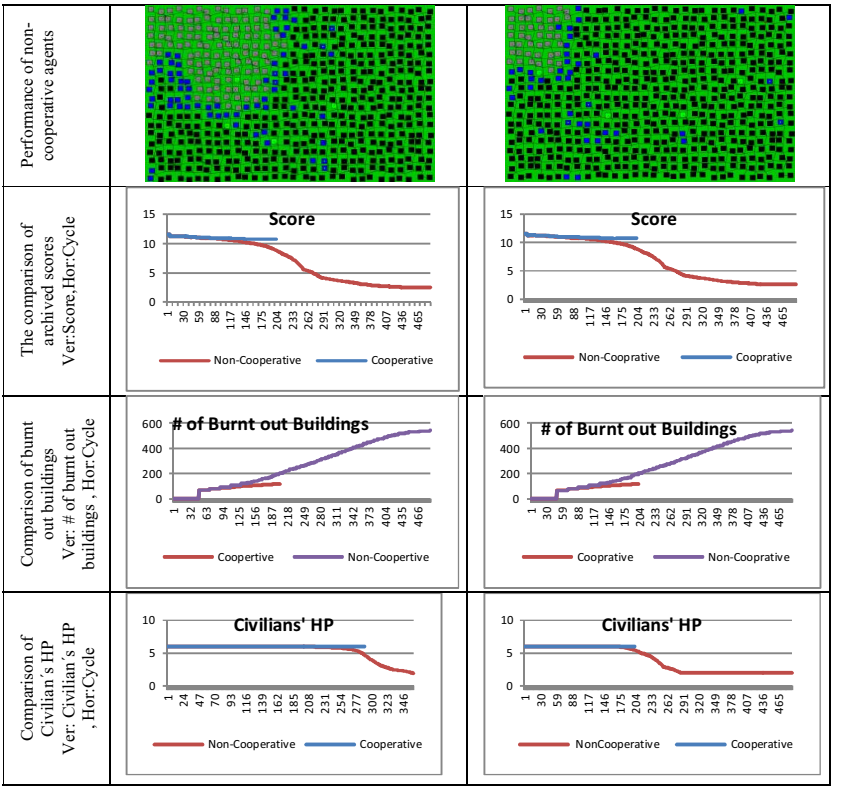

Fig. 5. (*continued*)

6 Conclusions

In this paper we discussed the use of temporal difference learning to find the optimum policy for fire extinguishing tasks of a group of firefighter agents, and results showed that agents trained by TD exhibit a good performance at extinguishing fires. This method has a good robustness and scalability.

Acknowledgments. The first author is supported by FCT under grant SFRH/BD/81155/2011. This work has been partially funded by FCT Project ACORD - Adaptative Coordination of Robotic Teams (PTDC/EIA/70695/2006).

References

1. Reis, L.P., Lau, N., Oliveira, E.C.: Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents. In: Hannebauer, M., Wendler, J., Pagello, E. (eds.) Reactivity and Deliberation in MAS. LNCS (LNAI), vol. 2103, pp. 175–197. Springer, Heidelberg (2001)

2. Stone, P., Veloso, M.: Task Decomposition, Dynamic Role Assignment, and Low-Bandwidth Communication for Real-Time Strategic Teamwork. Artificial Intelligence 110(2), 241–273 (1999)
3. Stone, P., Veloso, M.: Layered approach to learning client behaviors in the robocup soccer server. Applied Artificial Intelligence 12(2-3) (1998)
4. Kleiner, A., Dietl, M., Nebel, B.: Towards a Life-Long Learning Soccer Agent. In: Kaminka, G.A., Lima, P.U., Rojas, R. (eds.) RoboCup 2002. LNCS (LNAI), vol. 2752, pp. 126–134. Springer, Heidelberg (2003)
5. Jennings, N.R.: Controlling Cooperative Problem Solving in Industrial Multiagent Systems using Joint Intentions. Artificial Intelligence 75(2), 195–240 (1995)
6. Lekavý, M., Návrat, P.: Expressivity of STRIPS-Like and HTN-Like Planning. In: Nguyen, N.T., Grzech, A., Howlett, R.J., Jain, L.C. (eds.) KES-AMSTA 2007. LNCS (LNAI), vol. 4496, pp. 121–130. Springer, Heidelberg (2007)
7. Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.): RoboCup 2005. LNCS (LNAI), vol. 4020. Springer, Heidelberg (2006)
8. Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.): RoboCup 2006. LNCS (LNAI), vol. 4434. Springer, Heidelberg (2007)
9. Visser, U., Ribeiro, F., Ohashi, T., Dellaert, F. (eds.): RoboCup 2007. LNCS (LNAI), vol. 5001. Springer, Heidelberg (2008)
10. Kok, J.R., Spaan, M.T.J., Vlassis, N.: Multi-Robot Decision Making using Coordination Graphs. In: Proceedings of 11th International Conference on Advanced Robotics (ICAR), Coimbra, Portugal, pp. 1124–1129 (2003)
11. Paquet, S., Bernier, N., Chaib-draa, B.: Comparison of different coordination strategies for the roboCupRescue simulation. In: Orchard, B., Yang, C., Ali, M. (eds.) IEA/AIE 2004. LNCS (LNAI), vol. 3029, pp. 987–996. Springer, Heidelberg (2004)
12. Mohammadi, Y.B., Tazari, A., Mehrandezh, M.: A new hybrid task sharing method for cooperative multi agent systems. In: Canadian Conf. on Electrical and Computer Engineering (May 2005)
13. Martínez, I.C., Ojeda, D., Zamora, E.A.: Ambulance decision support using evolutionary reinforcement learning in robocup rescue simulation league. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006. LNCS (LNAI), vol. 4434, pp. 556–563. Springer, Heidelberg (2007)
14. Paquet, S., Bernier, N., Chaib-draa, B.: From global selective perception to local selective perception. In: AAMAS, pp. 1352–1353 (2004)
15. Amraii, S.A., Behsaz, B., Izadi, M.: S.o.s 2004: An attempt towards a multi-agent rescue team. In: Proc. 8th RoboCup Int'l Symposium (2004)
16. Abdolmaleki, A., Movahedi, M., Salehi, S., Lau, N., Reis, L.P.: A Reinforcement Learning Based Method for Optimizing the Process of Decision Making in Fire Brigade Agents. In: Antunes, L., Pinto, H.S. (eds.) EPIA 2011. LNCS, vol. 7026, pp. 340–351. Springer, Heidelberg (2011)
17. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)