RoboCupRescue System and Arian: A Flexible Infrastructure for Multi-agent Research and Education

Mazda Ahmadi
University of Texas at Austin
mazda@cs.utexas.edu

Tomoichi Takahashi
Chubu University, Matsumoto, Kasugai, Japan
ttaka@isc.chubu.ac.jp

Jafar Habibi
Sharif University of Technology, Tehran, Iran
habibi@sharif.edu

Tetuchiko Koto
University of Electro-Communication, Japan
koto@takopen.cs.uec.ac.jp

Abstract

One of the main requisites for multi agent researchers is a multi-purpose infrastructure that enables them to create, test and evaluate protocols, methods and algorithms. RoboCupRescue simulation environment is a flexible and configurable environment that can fill this gap. With changes in its configuration files, it will gain different features, so different methods can be designed, implemented, and tested. This environment is actually suitable for most multi-agent research issues such as cooperation, coordination, agent communication languages, and agent micro level issues. Sensible roles and tasks in the system motivate more research and make this environment a suitable infrastructure for educational purposes.

Arian is a set of implemented agents for RoboCupRescue system. Using Arian agents will speed up implementing agents for this environment.

In this paper we will introduce RoboCupRescue simulation environment and Arian agents along with some examples of research issues in this environment. We will also present some of RoboCupRescue sub-problems for specific research goals. Although RoboCupRescue can be considered as an application of multi-agent research, in this paper we will focus on the suitability of this environment as an infrastructure for multi-agent research and education.

1. Introduction

Recently, multi-agent systems have gained a lot of attention. Robotic soccer, intelligent control of multiple agents, automated driving, and e-commerce agents) are examples of challenging multi-agent domains.

Infrastructures help researchers to design, implement, and evaluate their methods. Some of these methods are domain dependent. Furthermore, domain independent algorithms can be designed and tested with the help of general infrastructures. RoboCupRescue simulation environment is an open source, flexible infrastructure. There are some defined roles and tasks in the environment. Roles and tasks from other environments can be translated to the defined tasks and roles in this domain. The problem that should be solved in this system (i.e. rescuing a city in disaster times) is a tremendous and broad issue. This problem can be divided into sub problems, each of which may also be required to be solved in different configurations of the environment in its turn (e.g. the range of agents' view can be configured differently, or the earthquake magnitude may change). Each sub problem with any of these configurations can be used as a research issue.

RoboCupRescue simulation and Arian's contribution to multi-agent society lies on two aspects:

- **Educational purposes:** The actions and states in this system are understandable to all educators. Agents are simulation of real humans with known tasks such as extinguishing fires or rescuing civilians. This makes RoboCupRescue environment an interesting environment for multi-agent education.
- **Test-bed for researchers:** Problems from different domains can be created, tested, and evaluated with the help of this infrastructure (e.g. task allocation problems, designing ACLs, coalition formation, etc.)

The RoboCupRescue project has started for the research of disaster mitigation, search, and rescue problems [11]. A competition on building the most powerful agents for RoboCupRescue environment initiated in 2001 with the name of RoboCupRescue league, at Seattle.

The rest of our discussion is organized as follows: after introduction, we introduce the RoboCupRescue simulation environment in section 2. In section 3, an overview of

Arian agents is presented. In section 4, different research issues such as coordination, cooperation, agent communication languages, etc., that can benefit from RoboCupRescue simulation environment is presented. Some sub problems in this domain that can be used as multi-agent research areas are presented in section 5. A comparison with other infrastructures is given in section 6 and finally, we will conclude the paper with section 7.

2. RoboCupRescue Simulation System

RoboCup Rescue Simulation environment is designed to simulate the rescue mission problem in real world[4]. In this simulation system a communication center and a number of simulators are existent to simulate the traffic after earthquake, road blockage, fire accidents as a result of gas leakage, etc[4].

RoboCup Rescue Simulation environment is a heterogeneous multi-agent system in which the agents correspond to the agents involved in a real rescue mission. Types of agents are as follows :

- **Rescue agent:** It is responsible for saving civilians and carrying them to the refuge.
- **Rescue center:** It is Responsible for organizing affairs between rescue agents. It also communicates and cooperates with other centers.
- **Fire brigade:** Responsible for extinguishing burning buildings.
- **Fire station:** It organizes the function of fire brigades
- **Police agent:** It is responsible for clearing roads and opening the traffic locks caused by the disaster.
- **Police station:** Like other centers, it is responsible for organizing affairs between agents.

In this system there exists a civilian simulator to simulate an ordinary civilian. The civilians do not possess a specific intelligence. The goal of the intelligent agents is to save civilians.

The important point in this simulation is the collaboration between these agents. This means that if agents in this simulation environment try to do their best independent of the other agents, a proper result could not be achieved. For instance, when a rescue agent should pass through a blocked road to save a civilian, it is the police agents' duty to clean the road[3].

All the modules have been implemented under Linux. Some in C++ language and some in Java language.

Although it is possible to run the system on one PC, optimal execution speed can be obtained by using 3 PCs, each having a 1 GHz processor and 256 MBs of memory.

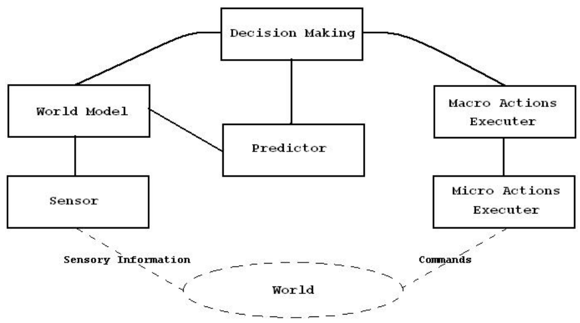

Figure 1. Arian agents' overall architecture.

For more information on the structure of the simulation and how simulation runs refer to the system manual /cite-manual [1].

3. Overview of Arian Agents

Arian was the champion of the RoboCup2002 and RoboCup2003 rescue simulation league in Fukuoka and Padova and won the second place in the RoboCup2001 in Seattle. The aim of this paper is not to answer the question why Arian won the RoboCupRescue simulation league championship, but to present the Arian library as a tool for building new agents[2]. The overall architecture of Arian agents is presented in figure 1. Each module is independent of the others and can be implemented separately. As we will see, the main benefit of using Arian as the basis of developing agents for RoboCupRescue simulation environment, is that the researchers can only design the decision making module. In the rest of this section we will briefly describe Arian architecture's modules.

3.1. Sensor

As mentioned before the communication protocol used in RoboCupRescue system is based on UDP. The sensor module facilitates communication between server and agents with the help of a simple interface. This module parses the received sensory information.

3.2. World Model

This module provide an interface for manipulation and extraction of high-level information from a world model

1 RoboCupRescue simulation system can be downloaded from *http://robomec.cs.kobe-u.ac.jp/robocup-rescue/download.html*

2 Arian agent codes can be downloaded from *http://ce.sharif.edu/~ arian/download.html*

database. This world model database is updated in the beginning of every cycle. The class hierarchy of objects in the Arian's world model is shown in figure 2. Each of these objects has the properties of the corresponding objects in the world.

3.3. Predictor

Given a macro action, a world state, and a prediction time, this module predicts the state of the world in the given time. This module works based on hundreds of previous runs and statistical methods.

3.4. Macro Actions Executer

A macro action is an action that lasts more than one single cycle. The macro actions in current version of Arian is as follows:

- **Extinguish a Building till totally extinguished:** In this macro action, the agent spreads water on the specified building with maximum power until the building gets extinguished.
- **Clear a long Road:** In the simulated world, each long road consists of a number of short roads. In each cycle an agent is able to clear only a short road, so clearing a long road takes more than once cycle.
- **Rescue a Civilian:** Rescuing of a civilian consists of several primitive actions that this module takes care of. First, the civilian should be taken out of the building he is in, then it should be transferred to an ambulance and the ambulance should take the civilian to the closest refuge and finally, drop him there.
- **Explore the City:** In many situations, agents tend to explore the city in order to get the most possible information about the world. This module handles this matter with some graph based algorithms.
- **Gotoxy:** By executing this macro action, the agent tends to move to another position that can probably take more than one cycle. There can be different algorithms for finding a path between two positions. In Arian agents there are two implemented algorithms for path planning, one based on the A* algorithm and the other on the Floyd algorithm. These methods get current and final positions and return a route between these two positions. Although these algorithms may not be optimal, but are useful for those whose research focus is not path planning.

3.5. Micro Actions Executer

This module handles communication with the kernel and sends commands to the kernel. For each of the legal commands that agents can send to the kernel, there exists an interface in this module.

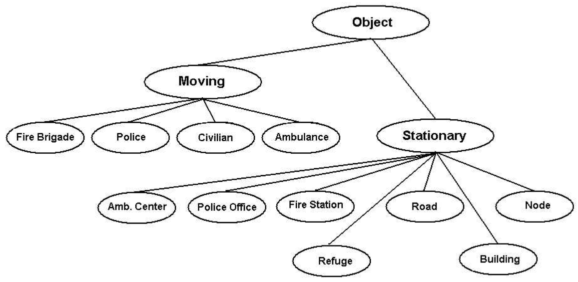

Figure 2. Objects' hierarchy in the world model. The nodes represent objects and two nodes are connected if the lower node inherits the upper one.

3.6. Decision Making

Usually this module is the one that most researchers are interested in designing and implementing it. The *decision making module* in Arian agents is complex and introducing it is not the aim of this paper. It also should be noted that it is possible for this module to directly call micro actions, although it is not used in Arian agents.

4. Research Areas

RoboCupRescue simulation environment is a complex system that has many sub problems. Because of many sub problems in this system, it is a multi purpose infrastructure for research in macro and micro issues in agent technology. In the following subsections of this section we will point out how the main issues in multi-agent research can be implemented and tested in this environment.

- **Cooperation:** Different aspects of cooperation such as protocols (e.g. [2]), formal models for cooperation (e.g. [9]) and coalition formation methods (e.g. [10]), are active research areas in multi-agent research and education. For all the mentioned aspects of multi-agent research issues, rescue simulation environment can be a useful infrastructure. To accomplish most of the tasks in the system, there is a need for cooperation. Two different types of cooperation can be distinguished, cooperation between homogenous agents, and cooperation between heterogeneous agents. Both are active research areas in RoboCupRescue domain.
- **Coordination:** Coordination methods and protocols are widely used in multi-agent systems (e.g. [8]). Coordinated actions for homogenous and heterogeneous agents is essential in this system. Need for coordinated

actions can be seen in different sub problems. Extinguishing fires need coordination between police agents and fire brigades. For example a simple protocol for this coordination, is that police agents clear the roads close to buildings which are on fire, since it is more probable that these roads are needed by fire brigades[3].

- **Agent Communication Languages:** Multiple research have been done on agent communication languages (e.g. [6, 1]). Having heterogeneous agents that can communicate in different situations with complicated and different types of communications, makes this system a perfect infrastructure for testing different agent communication languages.
- **Multi-agent planning:** Planning for accomplishing different tasks in the world should be done. Rarely a single agent can accomplish a goal by itself; so, multi-agent planning is also needed in this environment.
- **Multi-agent learning:** Different learning methods can be designed and used for accomplishing different tasks or for having an up-to-date world model.
- **Agent architectures:** Agents in this environment are heterogeneous and very complex. This variaty motivates designing new agent architectures.
- **Agent representation and specification formalism:** In this environment agents are simulation of real humans. A formalism for agent representation and specification is an interesting issue.
- **Rational actions:** Time is an important factor. When time goes on, victims die, fires spread and the state of the world changes drastically. In this situation rational actions are really needed.

5. Sub Problems

In this section we will present two sub problems of the whole system. These sub problems are broad and different research issues can still be used in each of them. There are some changes that can be done in configuration files in order to produce new sub problems (e.g different range of information, different number of controlled agents, different maps, etc.). These different configurations can be used in either of the sub problems.

5.1. Problem 1: Fire Brigades to Extinguish Fires

In this sub problem the only agents are fire brigades and possibly a fire station. Also the blockage simulator will not be running in order to enable fire brigades to move easily. The goal of fire brigades is just to extinguish fires and choose the best target with the aim of reducing the number of burnt buildings.

The priority of buildings lay on different parameters. For example when a neighborhood is on fire, it is wise to extinguish the boundary buildings. Another parameter can be seen in different building kinds (i.e. wooden, steel or concrete buildings). Wooden buildings burn and spread fire much faster than steel buildings. One of the important parameters in this system is the cost of agent movements. It takes time to move from a position in the city to another position.

5.2. Problem 2: Police Agents Help Fire Brigades to Extinguish Fires

In this problem fire brigades and police forces along with their centers participate in the action.

In this case some of the roads are blocked and moving from one position to another is much harder. Even there are cases in which there is no route between two places. Police agents should effectively clear roads in order to help fire brigades to move in their desired manner.

If you choose not to give complete information about buildings which are on fire to fire brigades, police agents can also help fire brigades to find the firing buildings.

6. Comparisons with Other Infrastructures

There are a number of generic infrastructures that help agent researchers to develop and test their methods and algorithms. A generic infrastructure is the one that the agents, groups and roles are not pre-defined in them. and the researcher can define agents, roles and groups as he prefers.

Some examples of these infrastructures are MadKit [5], Zeus and AgentBuilder. These infrastructures have different architectures, some allow various architectures but some force a specific architecture. Some only allow java agents while some allow all programming languages.

Although these infrastructures have differences between themselves but the main difference between these infrastructures and the RoboCupRescue simulation environment, relies on the fact that in RoboCupRescue simulation environment, there are some predefined tasks and roles (e.g. there is a fire brigade role and the task of extinguishing fires). The main advantage of generic infrastructures to RoboCupRescue is that in generic infrastructures problems from different domains can be transformed more easily to the domain. But in RoboCupRescue, roles in the real world should be translated to the defined roles in the system. Although mapping roles from other domains to pre-defined roles in RoboCupRescue is not hard, but it is not as simple and accurate as in generic infrastructures.

3 A more complex version of this protocol have been used in the Arian source code.

However as mentioned before, defined tasks and roles in the RoboCupRescue motivates new research fields. It benefits from sensible roles and tasks that make it convenient for multi-agent education.

In the rest of this section we will compare RoboCupRescue simulation environment with RoboCup soccer simulation environment that is also a good and well-known infrastructure for multi-agent research and education [7].

6.1. RoboCup Soccer Server and CMUnited Clients

RoboCup-soccer is a simulation of robotic soccer. It is a widely used multi-agent infrastructure [7].

The most important advantage of RoboCupRescue over RoboCup soccer and other domains is that it can be used for all kinds of multi-agent research issues. From research and educational perspectives, we can point out the following differences between these two infrastructures:

- Concerning macro level research issues such as cooperation, coordination and agent communication languages, RoboCupRescue is more general than RoboCup soccer. Although macro level research issues can be implemented and tested in RoboCup soccer environment, but limited number of agents and lack of agents' heterogeneity reduce its generality.
- Evaluation in RoboCup soccer is more complicated than RoboCupRescue. In RoboCup soccer playing 10 matches in a row with two identical teams of agents will have different results. Although there is randomness in RoboCupRescue, but the number of games for signification is much fewer than RoboCup soccer.
- RoboCupRescue is newer than RoboCup soccer, so the soccer simulation environment is more robust and there are more documents available about it.
- Soccer simulation is very easy to understand. Although understandability of RoboCupRescue is an advantage of it compared with typical multi-agent environments, but in comparison with RoboCup soccer, it would take more time to be understood.
- Soccer simulation is more interesting for young students. The excitement of real soccer can be seen in RoboCup soccer competitions.
- Using CMUnited clients in RoboCup soccer and Arian agents in RoboCupRescue, implementing agents in both of these environment are quite simple.

7. Conclusion

The RoboCupRescue environment and Arian agents are a flexible and general infrastructure for multi-agent research. Availability of lots of sub-problems enables this domain to be used for various research issues. It is easy to understand and design and implemention of agents in this environment is also simple.

For research purposes, most of the domain independent research issues can be translated to a task in this system. Tasks in this system can also help to design domain dependent methods and then transform these methods to other domains.

For educational purposes, it is a perfect infrastructure; since students can get familiar with it very fast and can easily design tasks to accomplish familiar tasks (e.g. extinguishing fires) that is very exciting.

References

[1] T. Balch and R. C. Arkin, "Communication in Reactive Multiagent Robotic Systems", Autonomous Robots, pp 27-52, 1994.

[2] M. d'Inverno, D. Kinny and M. Luck. "Interaction Protocols in Agentis", In Proceedings of the Third International Conference on Multi-Agent Systems, IEEE Press, 1998.

[3] J. Habibi, M. Ahmadi, A. Nouri, M. Sayyadian, M. M. Nevisi, "Utilizing Different Multiagent Methods in RoboCupRescue Simulation", In Proceedings of the RoboCup-2002 Symposium, Fukuoka, Japan, 2002.

[4] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjou, S. Shimada; "RoboCup Rescue: Search and Rescue in Large-Scale Disasters as a Domain for Autonomous Agents Research." Proc. IEEE Int Conf. on System, Man and Cybernetics (1999).

[5] MadKit - A Multi-Agent Development Kit, Laboratoire d 'Informatique, Robotique et Micro-Electronique de Montpellier (LIRMM), 161, rue Ada-34392 Montpellier Cedex 05 - France, May 2000 (http://www.madkit.org/).

[6] I. NODA and K. Shinoda, "Design of Communication and Language of Agents for Rescue Simulation", SICE SI 2001, pp.289–290, SICE System Integration Division, 2001.

[7] I. Noda and P. Stone, "The RoboCup Soccer Server and CMUnited Clients: Implemented Infrastructure for MAS Research" Journal of Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publisher, 2002.

[8] D. Pynadath and M. Tambe, "Team coordination among distributed agents: Analyzing key teamwork theories and models", Proceedings of the AAAI Spring Symposium on Intelligent Distributed and Embedded Systems, 2002.

[9] D. Pynadath and M. Tambe, "The communicative multiagent team decision problem: Analyzing teamwork theories and models", Journal of AI Research (JAIR), 2002.

[10] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and T. Tohm, "Coalition Structure Generation with Worst Case Guarantees", Artificial Intelligence, 111(1-2), 209-238, 1999.

[11] The RoboCupRescue Technical committee, RoboCupRescue Simulator Manual.