Impossibles08 Team Description
RoboCup Rescue Agent Simulation

Jafar Habibi, Soheil Hassas Yeganeh,
Moslem Habibi, Ali Malekzadeh, Amir Malekzadeh, Seyyed Hossein Mortazavi,
Hassan Nikaein, Mohammad Salehe, Mostafa Vafadoost, Navid Zolghadr

Sharif University of Technology,
Department of Computer Engineering
impossibles@ce.sharif.edu

Abstract.

The RoboCup International competition held annually has become a stage for those interested in artificial intelligence to come together and take part in this ever growing event. The RoboCup Rescue Agent Simulation league is one of the many existing leagues which the Impossibles team is participating in. This documentation presents a brief introduction to the Impossibles Team's efforts to find scientific solutions to the problems posed by the Rescue Agent Simulation. We have used various computer science algorithms in our code, including emotional decision making, world graph modeling, multi-criteria shortest path, and probabilistic theories. We have also adopted some heuristics to tackle intractable decision problems in Rescue Agent Simulation. These heuristics improved the performance of our algorithms while keeping the algorithms approximately optimal.

1 Introduction

In the Rescue Agent Simulation, robotic agents compromising Ambulances, Police Forces and Firefighters must save civilians and stop the destruction of a city hit by an Earthquake. Firefighters control the spread of fire in the map, Police agents clear blocked paths and the Ambulance agents rescue civilians from beneath the rubbles of buildings. The simulation must match real world limits and problems as accurately as possible, in the hope that eventually these agents can be used in real life to help in disaster areas.

The goal of the Impossibles team is further research into these areas and testing new and classical algorithms to get the results desired, also the team has in mind that Iran is one the world most earthquake prone countries and as the not so long ago earthquake in Bam showed, can cause massive economical and more importantly human losses. Our motivation therefore comes from the need our country has to counter such disasters and our hope to find a scientific solution for that need.

This paper is structured as follows. In the second section the high level software architecture of our framework is discussed. In the third section, Heuristic Mechanisms, we explain some common but important techniques used by

all agents, then in Decision Making section, algorithms utilized in the decision making process are presented. And in section 5, some technical details about the agents are presented.

2 Software Architecture

Each agent in our framework is implemented based on a set of base objects named *agent commons*. As shown in figure 1, the *agent commons* module provides a huge number of useful features:

- *Radar*: One of the most important features needed to implement a Rescue Agent Simulation team is *information gathering*. Any information gathered by the agents are broadcasted using the Radar module.
- *World Graph*: Using naïve data structures to handle world constructs (e.g. roads, buildings, nodes, etc.) is not a good choice. We have implemented the World Graph module that creates an *abstract view* of the world via a directed multi-graph, and also provides some useful algorithm implementations like shortest path and minimum spanning tree.
- *Offline Search*: At each time instance, some unexplored parts of the map may contain undiscovered civilians. The Offline Search module provides the mechanism for agents to search unexplored part of the map in the shortest time possible.
- *Disaster Environment Prediction*: For all decision making processes, an agent needs to know what will happen in the next cycles. The Disaster Environment Prediction module predicts the state of the world in next few cycles. It is also important to note that we can not have an accurate prediction because of the stochastic nature of the simulation server.
- *Emotional Decision Making*: In some cases, our normal rational algorithm can no longer provide an optimum outcome. Here, we have a tradeoff between saving some parts of the map or trying in vain to save the whole map. Here our Emotional Decision Making module give us some guidance as to which path to take when the system is out of control.

These modules are discussed in sections 3 and 4. To implement an agent one should use the base classes and also apply these common features. The agent specific ideas and algorithms are discussed in section 5.

3 Heuristic Mechanisms

We make use of many essential mechanisms which play an important role in our decision making, but are kept separate from the decision making process. Here we list some of the more notable ones.

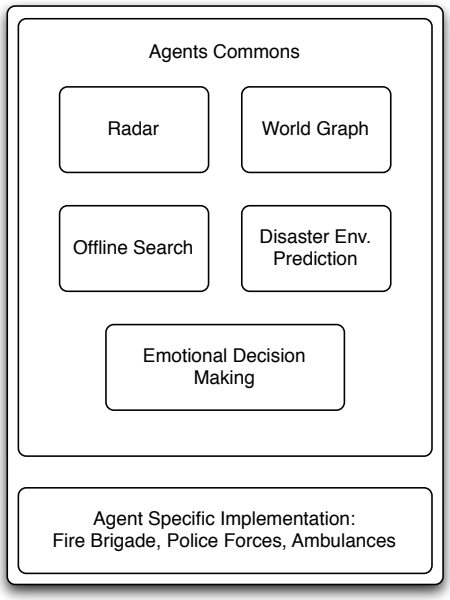

Fig. 1. Impossibles Software Architecture

3.1 Radar

Most of the information gathered by the agents is of what they sense in their surroundings, but many restrictions apply to this information gathering, for example each agent can only see a radius of 10 meters, which keeping in mind the large scale of the maps, is a pretty small section. Even this minimal information is only gathered at the start and end of a trip, so no information is gathered in between. Such restrictions mean that each agent can only act on information which is very partial and at most times out of date , which is not a very efficient way of making decisions. Our solution to this problem is that the information gathered by each agent is passed on to all agents, we call this the radar.

The agents interact with radar in three phases, in the first phase all agents send any new information to their respected headquarters. In the second phase the headquarters send this information to each other and in the last stage the combined info is sent to all. This may seem like a simple mechanism, but some serious problems arise which we have solved.

First of all, the three phases of the radar need the decision making to start with some delay so that decisions are made on complete information. These phases need to be refined so as this delay is minimal. Second, the information that needs to be send is usually more than the limit an agent can fit into a tell command. Two solutions exist for this problem, first of all only information about events which have changed from what is kept in our world model are sent

(New Info), secondly headquarters must merge all the new information they receive before sending them, so as repetitive information is not sent.

This creates many possibilities for the team, also keep in mind that this mechanism acts at the start of each new cycle automatically and is completely separate from the decision making. If the headquarters are inactive (as in the maps lacking centers) then each of the three agents types create their own individual radars, so three smaller versions of the radar system are created and used by each agent type. Also at a specific time and place representatives of each agent type meet to give their radar information to each other so all their radars are updated. We call this the rendezvous tactic.

3.2 World Graph

As described before we have modeled our world using a graph called the World Graph. This abstraction of the world has posed many challenges in its implementation but the overall results have benefited many aspects of our agent's decision making process. Our main problem was that for the World Graph to be useful we had to store it in many different formats, as each use of the World Graph required a different representation, and therefore any changes made had to be updated many times. The World Graph was mainly used to decide the best and shortest method to reach a destination from a source node and to do this we tested an assorted number of graph algorithms at the end relying on a combination of some of them. After the readiness of the World Graph for general use we saw a huge improvement in the effectiveness of all agents and path traversals were reduced to a minimum.

3.3 Disaster Environment Prediction

One of the best approaches to deal with the disaster which has taken place in the simulation is to predict what events will unfold in the next few cycles. We usually encounter this problem in separate parts of the decision making mechanisms. Our idea is that by collecting these predictions and placing them in the proper structure we can get a good sense of what our world model will be like in the near future and prevent repetitive tasks being performed. The most important mechanism we have develop in order to predict the disasters, is our *fire simulator*. Fire simulator is an approximate fire spread predictor that uses all the information that radar has been collected. Using the fire simulator all the agents can have a good approximation of the temperature of a building.

By concentrating on this system we can utilize it to use prediction techniques with maximum efficiency and also find a mechanism to learn how disaster events spread in the city.

3.4 Offline Search

Keeping in mind that our final goal is saving civilians, scouting unexplored parts of the map to find trapped civilians is of the outmost importance. If we place our

agents in the smallest collection of points from which the whole map is visible we can scout the map in the shortest time possible.

This is a dominating set problem and because it is a NP one, we need to use approximation algorithms to finds its answer online as the game is being executed. For maps explored before (the major maps) we can find these points in advance, and use their locations during the game. So if all the important points of re-con are found the whole map is visible to our agents, and make them able to optimally rescue the civilians.

4 Decision Making

4.1 Decision Making Architecture

We are trying to use a three layered decision making system. This decision making system has a tree structure with three levels or authorities which works in normal and emergency situations. Agents are leaves of the decision tree, parents of each group of leaves are their corresponding headquarters, and root of the tree is a coordinator who has the greatest authority. These are headquarters that make general decisions and specify the general duty of their agents. Now, agents of each headquarter must decide themselves and find the best way that they can perform command of their office.

In order to make it obvious, consider police forces and their headquarters. Headquarters of the police forces command each team of them to go to specific regions of the map for unblocking the blocked roads. Now each group of police agents should find a good way to those places and when arriving to the mission regions they should find a good solution to do their operations. Note that these teams can have a commander from themselves which lead team members for some jobs and control them at some operations. There are some times that different kinds of agents do not make a good teamwork, they are not supported by the others and all of the agents do not follow a suitable program. At these times we need a coordinator whose job is to make different groups work together in an organized way to achieve the goal and make the whole job effective. Consider that the coordinator decisions are strategic and their frequency is lower than headquarters.

4.2 Priority

During the simulation each of the rescue agents have tasks they must perform which are spread throughout the city. Agents must decide which of these tasks should be carried out first, this has a significant impact on the rescue process and can even be considered as one of the most vital sections we need to consider.

Keeping in mind the whole disaster situation we can use Central Processing and Distributed Processing or a mixture of both these methods to specify the importance of each task for the agents. We are trying to assign the tasks to the agents in the best possible way, and in this process we must keep in mind the

importance of the task, its duration and the time taken to switch tasks. Also it is important to note that a task's importance changes as time passes.

The algorithm used for this purpose is centralized and its results are sent to the agents. Each agent must perform the task that is assigned to it. This assignment problem is pretty complicated and can be considered as a case of the assignment algorithm in graphs. So to solve it in an acceptable time we must use approximation algorithms. Studying recent researches done in this field we saw a resemblance between this problem and the Bicriteria[1] and Steiner tree[2] problems. By modeling our problem using discrete mathematics and using papers published in these fields we found an acceptable and approximate algorithm that fulfills our needs.

It is important to note that, the mechanism in which the tasks are assigned is changed in the lack of headquarters. In these cases, the agents decide independently on their own, so they may decide to work on the same task at a time. To avoid these difficulties, we have embedded a priority mechanism in our agents. In duplicate decision scenarios, the agent with less priority will change his decision.

4.3 Emotional Inspiration and Emergency Decision Making Approach

Recent neurological findings indicate a vital role emotions play in decision making, understanding and learning. Emotions affect most of rational thinking mechanisms [3]. Among different emotions, stress is a special one and its influence on the mental functionality and physical and mental performance is not ignorable.

Based on the various researches done on the functionality of stress, decision making in emergency situation is completely different than normal and business decision making. In normal situations, decision maker considers different options and choices, analyzes strength and weakness of each of them carefully and finally makes up his or her mind. In emergency situations, because of time pressure, he selects the best way immediately based on his experiences. In a study of decision making by fire ground commanders, it is reported that they were not making choices considering alternatives or assessing probabilities. They saw themselves as acting and reacting on the basis of prior experience[dma]. Therefore, there is a possibility to decide correctly and increase system performance by experience and its retrieval in emergency situations. Such decision making process is called naturalistic decision making (NDM) process[5] which is different than the analytical decision making process practiced by business managers[6].

In this project we have introduced an emotional structure for decision making. The new structure is constructed by adding decision making module which operates under stress, to the old system which operates in normal situation. The module operating in emergency situations will control the system when it is stressful and by using its past experience does what had been done at that time.

The proposed idea can be briefly explained as some modules which are added to the previous decision making system enabling system for decision making in emergency situations. These modules are Stress sensor module and Emergency

decision making module which the latter has two sub-modules: Similar past experienced case exist and Revise the solution based on reusing the past case. Also, there can be some optional modules: Evaluate proposed solutions and Retain current case. You can see the proposed structure in figure 2.

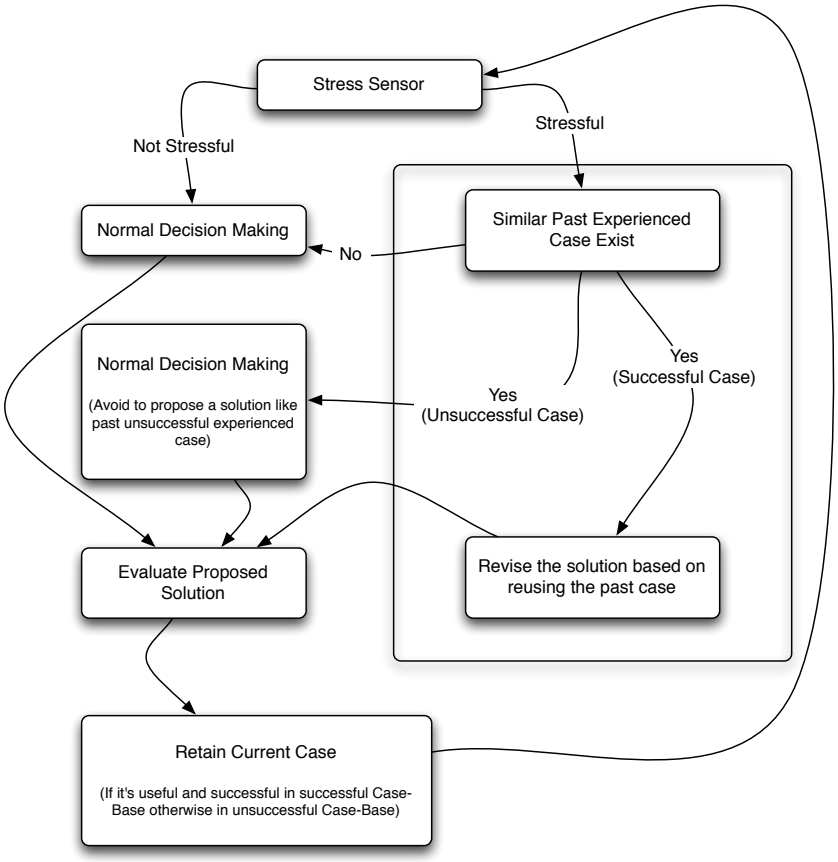

Fig. 2. Structure of proposed framework for emergency decision making

Stress sensor module has a stress detection function inside and indicates whether the situation is stressful or not. If it was not stressful the system derived to old decision making module and decides in normal situations. If it stressful, base on existence of a similar past experienced case the system will decide what to do. In emergency situations, system searches for the most similar past experienced case. If there is a case but it is an unsuccessful one, again system will lead to the normal decision making module and use its old decision making module with regard to this matter that the system attempts to avoid proposing

a solution similar to the solution of that unsuccessful past case. If there is a successful past experienced case, the system will use this experience to propose a solution for the current problem. This is done by Revise the solution based on reusing the past case module. The module considers the current situation and the situation of the past similar case, then adapts the past solution with regard to these differences to the new problem solution, and finally proposes a suitable solution for the new problem. This proposed solution can be evaluated as a good or bad one by Evaluate proposed solution module. This module should be accurate and its mistake can destroy the functionality of the system. If a good solution is recognized as a bad one or vice versa it will cause many problems. Proposed solution for the current problem can be retained in a case-base for future use. This can be done if there is a way for evaluating the solutions. If the solution evaluated as successful and useful, it saved in successful case-base for future usage and otherwise it will be retained in unsuccessful case-base for avoiding proposing a solution like this later. Runtime retaining the solutions has its own matters.

The results till now show undoubtedly a better performance than when the old system used just its normal decision making module. Also, the novel system could tolerate fault and resource failures in real world and simulation environment, and maintain performance at a specific level while crises especially for unpredictable conditions.

This brand new method of thinking has two main advantages in comparison with current decision making systems.

The first important advantage is fast and confident decision making while crises occur. In crisis occurrence, in order to reduce expanse of the disaster there is a vital need to decide rapidly. Although the decision made in this situation is general and is not necessarily the best one, we are sure that it not late and the disaster is not uncontrollable. Also, we are sure it has a degree of correctness in this critical conditions and it has been tested successfully in the past. This assurance is very important in emergency situations. In emergency situations a wrong decision can make disaster uncontrollable, so risking is not a wise decision.

Second advantage is fault tolerance for unpredictable and unforeseen probable failures. It is a natural characteristic of this structure. In crises, it likely that many resources became damaged, inaccessible or failed. System working under these situations had to be designed and implemented in a way that it does not crash with these failures and could tolerate lack of resources like inaccessible memory, low memory or low performance of processors. While there are enough resources, the proposed system works in normal and emotional modes according to the conditions. We may or may be we don define resource failure as stressful situations because they are forgotten, unforeseen, unpredictable, etc. In both of these conditions, predicted or unpredicted probable failures, the system will tolerate the fault as follows. If some resources fail, system will sense stress due to the conditions defined for it or the decrease of performance and functionality of the system cause stressful situation. Therefore, automatically the emergency decision making controls the events using its past experienced

cases. This way of decision making needs resources less than the previous one and thus the performance will increase and maintain in a specific level. The performance of the system in these conditions is not as high as the performance in the normal situations but the available resources are not as many as normal situations too. This performance is acceptable with regard to the resource failures we faced. When the situations became controlled and it no more stressful, system attempts to change its way to normal decision making. In these conditions, if resources are repaired and now there are adequate available resources, the system performance improves due to its normal decision making and if still there are inadequate resources the system will return to the emergency decision making. By this, the performance of the system will maintain at a specific level and could adapt itself with different situations. It more important that even if these critical situations and resource failures have not been predicted before and are unknown till occurrence, the system could adapt itself with them.

5 Agents

The most important part of all Rescue Agent Simulation teams is the agents. In this section a brief discussion about all agent types (Fire Brigades, Police Forces, and Ambulances) is given.

5.1 Fire Brigades

One of our ideas for implementing the firefighting agents was to model their behavior based on read firefighters, therefore we needed a way to simulate human knowledge and behavior in our algorithms. For example in forest fires when the fire cannot be put out firefighters try to contain it by cutting of the trees adjacent to civilian areas so that it cannot spread to them, our agents too simulate such behavior and try to contain the fire when it seems it cannot be extinguished. This is done by dividing the map into different sectors, with each sector having the property that fire spread rapidly in it. When the firefighting agents feel that the fire cannot be put out they do not try to stop the spread in a sector but concentrate on the fact that the fire should not spread to other sectors.

Another point of interest is identifying the areas surrounding fires. Here we try to change the contour algorithm which is used in image processing so that the high cost of updating these areas is reduced and also so we can be able to identify the areas in a fire set that have not caught fire yet. For last years game we tried to identify the fire surrounding areas using buildings which have just caught fire and pre-processing on the map. Each burned building destroyed the viewing angle of some unburned buildings so the area around a fire can be easily recognized. Using this recognition our simulation became more real, for example in a forest fire no firefighter tries to put our a tree surrounded by burning trees.

5.2 Police Forces

In Robocup Rescue the police agents are helper ones. Police has the main goal of opening blocked roads and making sure that the other agents (ambulance and firefighting agents) can reach all parts of the map. The most important factor in implementing the algorithm for the police agents is how we prioritize the different tasks. These task, such as opening a path the the refuges, maintaining access to the fires, providing a means for the ambulance agents to reach buried civilians and agents and making sure that other agents and the civilians are not blocked, is done after evaluating the priority of each task and then are given to each police agent depending on its situation (what it is doing at the moment and its whereabout in the map) The police agents translate each task assigned to them as one which requests a series of blockades that must be cleared from a path. In our code, the Impossibles team, a layer is added to the tasks requested and the tasks assigned. In this layer the map is divided into different sectors. Such a sectored is designated an open sector if the agents can move in it freely. Are goal is to expand these already large sectors. Closed sectors also exist and are sectors which cannot be reached from an open sector but if by opening a path to one part of these sectors the whole sector can be accessed. We call a closed sector important if it is critical, like when there is a fire in it or some civilians are buried there. In these situations the police are tasked with linking a open sector to a closed one and also linking the open sectors together. One of the most important aspects of our algorithm is evaluating a task importance and assigning it to agents based on their whereabouts. In this situation we may assign to a police agent in limited time several task which are in the same vicinity. We call this problem priority assignment. Lets define it formally.

Definition 1. *(Priority Assignment Problem) Several missions and commanders are spread throughout the map, we want to assign a subset of the missions to each commander, also keep in mind that the subset missions are given with certain priorities. We have the distance between each mission and also between a mission and a commander. Also each mission in itself takes a certain amount of time to be completed by the agents. The goal is minimize the time it takes to finish each mission*

Solutions to this problem is has been implemented in two ways:

- *Genetic Algorithms*: First we give a completely arbitrary assignment and in each subsequent step try the improve our assignments. And assignment is improved if the path of the commander with the largest path to take is minimum. In this solution we use the basic methods of genetic algorithms, like mutation and crossover.
- *Weighted Matching*: Our priority assignment problem is a sort of matching algorithm, the use of this method is expected. Here for the first mission of each commander we decide the most appropriate one and then as a commander changes it position to reflect the mission, we decide the second mission for it and so on.

After implementing each method we came to the conclusion that using genetic algorithms give us less performance but may give a better answer therefore it is suitable for a non-distributed environment whereas the weighed matching method is suitable for a distributed environment. At the beginning of the game most of the map is unexplored, therefore when an agent does not have a task it starts exploring the map. When using this ability, called map search , it is important that explored parts of the map are not explored again. Also we must make sure that several agents do not search the same region. With the help of our radar system, this feature has been implemented fully in our code.

5.3 Ambulance

At first we tried a greedy algorithm where we assigned to each ambulance a trapped civilian that seemed appropriate at the moment based on such factors as the civilians damage, estimated time of death and other factors. The problem with this approach was that we had manually entered the values used in the assigning process and therefore they were based upon our own estimates and did not always perform well.

So that the important factors that were needed were more accurate and their coefficients properly reflected their importance relative to each other, we used learning to come up with a new assignment algorithm. Therefore at each cycle the ambulances that are free choose a high priority job not yet taken from the job pool using the said algorithm. We also added the possibility that when an agent working on trying to save a civilian feels like it need help it sends a message to its center asking for backup and the center chooses a free agent or one that is working on a low priority task and reassigns it to help the needy agent.

6 Future Works

The Impossibles team intends to implement fresh ideas into it decision making after completely finishing the above mentioned parts and tuning them for maximum performance. For example some of the decision making algorithms are weight based and using reinforcement learning can have an immense effect on their results.

Also we have to improve our decision making process when no headquarters are available. The performance of our decision making framework degrades when the agents make decisions in a distributed manner and autonomously. We will improve the agent decision making at time when suffering from lack of global information.

Also, some part of our implementation should be changed (both in technical and algorithmic manner) to improve the overall performance of our framework.

References

1. Horst W.Hamacher, Stefan Ruzika, Stevanus A.Tjandra: Algorithms for time-dependant bicriteria shortest path problems

2. David S. Johnson, Maria Minkoff, Steven Philips: The prize collecting Steiner tree problem: Theory and Practice
3. Damasio, A.: Descartes Error: Emotion, Reason, and the Human Brain. Rockefeller University Press, New York, United States. 1994.
4. Klein, G., Orasanu, J., Galderwood, R. and Zsambok C.: Decision Making in Actions: Models and Methods. Ablex Publishing Corp., New Jersey, United States, 1993.
5. Klein, G.: The current status of naturalistic decision making framework. Decision Making under stress: Emerging themes and applications, R. Flin, E. Salas, M. Strub and L. Martin, (Eds.), pp. 13-25, Ashgate Publishing Ltd, UNITED KINGDOM, 1997.
6. Orasanu, J.: Stress and naturalistic decision making: strengthening the weak links, In Decision making under stress, R. Flin, E. Salas, M. Strub and L. Martin, (Eds), Ashgate Publishing, United States, pp. 43-66, 1997.
7. Habibi, J., Fathi, A., Hassanpour, S., Ghodsi, M., Sadjadi, B., Vaezi, H., Valipour, M.: Impossibles Team Description, 2005.
8. Habibi, J., Nowroozi, A., Boorghany Farahany, A., Habibi, M., Hassas Yeganeh, S., Mortazavi, S.H., Salehe, M., Vafadoost, M.: Impossibles 2007 Team Description, 2007.