RoboCupRescue 2010
Rescue Simulation League Team Description
IAMRescue (United Kingdom)

Francesco Maria Delle Fave, Heather Packer, Oleksandr Pryymak,
Sebastian Stein, Ruben Stranders, Long Tran-Thanh, Perukrishnen Vytelingum,
Simon A. Williamson, and Nicholas R. Jennings

School of Electronics and Computer Science,University of Southampton, Southampton, SO17 1BJ, UK

Abstract. In this paper, we outline the different software agents that are used in IAMRescue team for the Robocup Rescue Simulation League 2010. Specifically, we present a number of novel strategies that help our agents coordinate effectively within and across their respective ambulance, fire brigade and police force teams. For ambulance teams, we developed a dynamic scheduling mechanism for rescue operations. Furthermore, we designed police forces that strategically coordinate to remove blockades based on priority levels and effectively explore the space. Finally, we designed our fire brigades to contain fires within blocks of buildings and prevent them from spreading to the more vulnerable blocks where there are more civilians at risk. The interplay of these different agents allows IAMRescue to be very efficient in managing fires, clearing roads and saving civilians.

1 Introduction

Today, there is considerable endeavour in the domain of disaster management for distributed, agile and autonomous response in environments where uncertainty, scarcity of resources and bias are endemic. The Robocup Rescue platform addresses some of these issues in this domain. In particular, the aim is to design effective heterogeneous agents that manage the behaviour of ambulances, fire brigades and polices. The challenges here are multi-fold and, more specifically:

- To develop a distributed system architecture which operates effectively in the uncertain and dynamic environment of Robocup Rescue where communication, and thus coordination from a centre, is constrained.
- To design agents which collectively plan to achieve common goals.
- To design effective behaviours that ensure desirable overall properties (as an emergent behaviour of agents' local decision-making based on incomplete, imperfect information).
- To devise techniques to allow agents to effectively balance acting and in-formation sensing.

To address these challenges in the Robocup Rescue domain, we developed the IAMRescue strategy, which defines the behaviour of a number of different types of agents, including the ambulance team, fire brigade and police force agents, as well as their respective centres, which are required to facilitate communication between agents. The novel contribution of our work lies in the main decision-making processes for each of our agents, which are based on sound, principled techniques and algorithms from the areas of artificial intelligence, distributed optimisation, search and scheduling. In drawing from these established fields and extending them, our team is not only able to perform effectively in the competition, but also applies and advances the state of the art of the research community in the context of a real-world problem.

Against this background, we now describe in detail how we design these different agents to address the challenges outlined earlier on for effective disaster management. We will concentrate on a number of main components:

- Decision-Making: the high-level strategic behaviour of the agents.
- World Modelling: how we model the agents' view of the world.
- Coordination and Communication: how the agents communicate with each other (given a limited communication capability) for coordination.
- Disaster Prediction: how the agents predict disasters (e.g. through simulations).
- Routing: how the agents route themselves around the world.

In the following sections, we will describe in detail each one of these and how they contribute to the effective behaviour of the IAMRescue strategy. First, we consider the tactical part of the strategy which is the high-level behaviour of ambulances, police forces and fire brigades.

2 Decision-making: Agent skills and action selection

In order to be able to encode the agents' strategy in a way that makes it possible to alter the team's aims and objectives at different levels (e.g. at the level of goals, plans, and actions), we decided to divide the decision-making processes into three principal layers which are responsible for making decisions at varying levels of abstraction:

- The strategic layer is concerned with making high-level decisions about which goals a larger team of agents should pursue. Such decisions include which burning buildings to extinguish, where to unblock roads, or which groups of civilians to rescue next.
- At the tactical layer, these high-level goals are translated into a specific plan for a set of agents, and hence a specific task for individual agents. For example, given the goal to extinguish a number of buildings on fire, this layer allocates fire brigades to the buildings, based on factors such as their location or intensity.
- The operational layer is finally responsible for realising an individual agent's task as a sequence of atomic actions. This includes low-level decisions, such as finding the optimal route to a destination, deciding where to aim water jets and when to re-fill water tanks.

This hierarchy has been used successfully in disaster-response applications to abstract and coordinate decision-making within multi-agent systems [1]. We employ it here for a number of reasons. First, it provides a natural way of building well separated and modular decision procedures, where the detailed realisation of high-level decisions is left to progressively lower levels (which, in turn, do not need be concerned with the reasoning to arrive at these decisions). Second, the layers translate well to the distributed nature of the Robocup Rescue domain, where some team decisions can be delegated to a centre, but where most low-level decisions need to be taken by individual agents that have their own view of the world. More specifically, in our model, a centre is typically responsible for the strategic and tactical layers, while platoon agents make decisions at the operational layer after receiving tasks and goals from their centre.

In the following sub-sections, we discuss in more detail the decision-making procedures that the different IAMRescue agents adopt. Specifically, in Section 2.1, we describe our ambulance strategy. Then, in Section 2.2, we outline the strategy of fire brigade agents, and in Section 2.3, we cover our police force strategy.

2.1 Ambulance Team Strategy

The main task of ambulance teams is to determine how to best save victims trapped in collapsed buildings. To this end, we employ heuristics in the area of dynamic real-time scheduling. In more detail, we will need to determine for each victim i a deadline $t_d^i \in \Re$ and the optimal number of ambulances $N_{opt} \in [0, \infty]$ it needs for it to be saved by the deadline (except if it is not going to die by the end of the game). Now, each of these victims (or tasks), arrive in real time and may have changing deadlines (e.g. due to the building, in which it is, catching fire).

To deal with this, we follow a distribuited game theoretic approach from [2] which allows for ambulance agents to make local decisions and consequently reduce the amount of communication required. In more detail, we approximate the allocation problem described above using a series of static *potential games*, and then use a decentralised method for solving the approximating games that uses the distributed stochastic algorithm. Our algorithm is run by each individual agent and proceeds as follows:

```
for all $i \in A$ do
  Estimate marginal contribuition to the task of saving a particular civilian $t_d$
  Assign ambulance $i$ to civilian with highest marginal reward $t_d$.
end for
```

Because the underlying problem is a potential game, a greedy approach in the local problem will improve the global solution. With such an approach, there are a number of issues that could lead to computing sub-optimal solutions (which we do not elaborate here due to space restrictions). Furthermore, it relies on several estimates that are sometimes imprecise. To address these issues, we are currently working on several improvements to the ambulance strategy:

1. Perform a better allocation by computing the trade-offs in choosing an agent with an earlier deadline as compared with a number of agents with longer deadlines. This would avoid the agents wasting time to save one agent as opposed to working in parallel or together on other agents. In so doing, it would be possible to increase the number of agents that can be saved.
2. Improve the travel time estimation - with a better routing algorithm it will be possible to compute the approximate time it would take to route to a given victim. This would take into account blockages and traffic jams as well.
3. Improve multi-agent teamwork - it would be possible for ambulance agents to coordinate with fire brigade agents by sending them the locations of the victims they aim to save and, in turn, fire brigade agents would protect these locations from fires that are spreading.
4. Reduce need for communication by employing techniques such as [3] to allow agents to better solve the global problem on their own.

Our ambulance strategy worked very well on small maps during the 2008 competition. However, on bigger maps our agents were crippled by high memory requirements of the routing module. It is therefore important to ensure that the routing module of future versions of our agents are memory efficient and computationally efficient as well since ambulance agents need to compute expected travel times and routes a large number of times in order to compute the optimal allocation.

2.2 Fire Brigade Strategy

The purpose of fire brigades is to put out fires. However, due to scarce resources, it is usually not possible to put out all fires. In such a case, it is vital to prioritise fires with respect to regions within the world, and if these fires cannot be put out, we then need to minimise their damage by containing them.

We begin by first defining a graph that models how fire can spread between buildings. Every node represents a building and every edge represents a heat transfer channel. By doing this, we aim to model the dynamics of fire, and use this for the fire brigade strategy. Once the graph is built, we further define clusters of buildings (i.e. nodes). The rationale of dividing the world into clusters is that fires spread faster through clusters (given the proximity of buildings) rather than across clusters. Next, we define a fire site as a group of adjacent buildings that are on fire. Note that these buildings might be part of different clusters and might be separated by roads. Essentially, a fire site represents the extent of a fire as it spreads from its initial point. When two or more fire sites meet, they are combined into a single fire site. The perimeter of the fire site is essentially a set of buildings on fire on the perimeter of the fire site which threaten to burn adjacent buildings.

Against this background, we now detail how we prioritise clusters and fires from a fire brigade's perspective as ordered below:
Blocks:

1. The number of civilians and the criticality of their injuries are the foremost reasons for protecting a cluster first.
2. Because fires spread in all directions, a fire site at the centre of the world is more dangerous than one on the perimeter of the world, since the fire is likely to spread to more clusters adjacent to it. Thus, the more central fire sites are, the more important they are.
3. Clusters that are far from a fire site are less important.

Fire sites:

1. Fire sites with more clusters that are more at risk (as described above) are more important.
2. Fire sites that are closer to the fire brigade are given more consideration.
3. Fire sites that are more central (based on the same intuition as with clusters) are more dangerous and should be prioritised as such.
4. Small fire sites are easier to put out (preventing them from spreading, such that they need not be contained). Thus, we prioritise small fire sites over larger ones which generally need to be contained and cannot be put out completely.

By giving different weights to these priorities, we are able to effectively decide first, which fire site will be more damaging in the future, i.e. in a number of time steps (see Section 4 for more details of our disaster prediction techniques). Given this critical fire site, the aim is to contain the fire, preventing it from spreading it to the more important clusters. Next, we consider the clusters that are adjacent or contain the perimeter of the critical fire site. By considering the priority of these clusters, we determine which is the most important cluster to protect as the critical cluster. To effectively contain the fire from spreading to that cluster, we consider putting out the burning building on the perimeter that is closest to the critical cluster, defined as the critical fire-building.

Now, because of the local decision-making constrains of our agents, based on their individual belief of the world, they all locally determine a critical fire-building to put out. Specifically, at each time step, a fire brigade calculates a critical fire site, a critical cluster and finally a critical fire-building. Due to uncertainties in the world, they do not necessarily calculate the same critical building. Thus, there is usually an emergent coordination effort to extinguish these burning buildings, with some fire brigades often breaking off to focus on other (possibly closer) fire sites.

As future work, we intend to have more explicit coordination among fire brigades (with the fire centre coordinating the efforts) with a reasonable communication over-head. Furthermore, we intend to have more coordination with ambulances (with the ambulance centre coordinating with the fire centre). This would allow us to better prioritise clusters. In particular, ambulance efforts to save civilians in a cluster would increase the priority of that cluster. This would allow the fire brigades to slow down a fire sufficiently for ambulances to save civilians. Finally, more work will go into managing priorities of clusters and fire sites which essentially determine the behaviour of fire brigades.

2.3 Police Force Strategy

The aim of the police agents is to keep the roads clear from blockages, in order to allow the other agents to execute their strategies. Clearly, doing this quickly and efficiently is essential for the performance of the team, especially as most agents are initially highly restricted in their movement or even completely blocked. To address this, the strategy of the police agents includes two parts — first, they ensure that all agents are part of a fully connected road network, and second, they clear important roads on the map, to allow faster access to strategic locations. We describe both parts in detail in the following.

Initially, the police agents give priority to ensuring that all agents have access to the refuges and to fire sites. To do this, the police team creates a list of goals that need to be completed:

- Each police agent must have a clear path to a refuge.
- Each other platoon agent must have a clear path to a refuge.
- Each fire site must have a clear path to a refuge.
- Each refuge must have a clear path to every other refuge.

The goals are processed in the order shown above. Once they are completed, every agent, fire site and refuge is then guaranteed to be on a fully connected road network, allowing all agents access to critical positions on the map. For efficiency, these goals are allocated to agents such that the overall time spent on clearing roads is minimised. In more detail, we estimate the time it would take each agent to complete each goal and then employ the well-known Hungarian algorithm to allocate agents to goals in an optimal manner. As most of the knowledge about the world state is shared through communication, this is done locally in a decentralised manner in order to save time and bandwidth.

Once all the above goals have been achieved, the police agents start clearing strategic roads on the map, in order to improve access to important locations. This is done by attaching a value to each blocked road, which depends on two factors: (1) whether the block is generally in a highly-visited location (e.g., near a refuge or along a main stretch of road) and (2) whether it is currently on the shortest path between a refuge and an object of interest (e.g., a civilian or a fire site). To reduce the computational burden for this strategy, police agents do not coordinate explicitly when selecting roads, but rather each agent adjusts the value of a road depending on its proximity and then greedily clears the most important road that is not already being cleared by another police agent. This ensures that agents generally clear different roads in their current vicinity.

This concludes the outline of our ambulance, fire brigade and police strategies. In the following, we turn towards more general decision-making procedures that are shared by most agents. We start by looking at the world model in the next section.

3 World Modelling

The world model of each agent represents a view of the current state of the world, including highly dynamic information about the conditions of buildings, the locations of civilians and the states of other agents. As such, it is an integral part of each IAMRescue agent and provides the basic input data for our decision-making procedures.

At the most basic level, our world model stores the direct observations that an agent makes each time step. However, as this is clearly limited to the immediate surroundings of each agent, we use radio messages to relay observations to an agent's respective centre. This, in turn, filters redundant observations, collates the information and then broadcasts this to all agents again. Although introducing a time delay, this process allows each agent to build up a global world model rather than a limited local view.

In addition to these basic observations, we perform some reasoning on the world model to infer further knowledge. For example, we group adjacent buildings into larger blocks or several consecutive roads into longer main roads. This allows some of our decision-making procedures to work more efficiently and at a higher level of abstraction (see Section 2.2 for an example).

Now, in order to build and maintain a comprehensive world model, we explicitly include some monitoring and information gathering actions in the operational layer of each platoon agent (regardless of its type). These activities usually become active when no tasks of higher priority have been received from the centre, or when a new simulation run has just started. In particular, at the beginning of a run, all agents start by searching the map. This means going into every building that is not known. In so doing, agents can make use of a number of cues available to them in order to enhance the search for a victim or fires. For example, if an agent hears a victim calling for help and if it has never been heard before, the agent needs to search all buildings that are within its hearing distance. Also, by partitioning the map using the k-means clustering algorithm, we ensure that agents search an equitable and disjoint part of the map.

Given the basic world model, we now proceed to discuss how our agents predict future developments in the world and estimate hidden simulation parameters.

4 Disaster Prediction and Parameter Learning

Disaster prediction and parameter learning are essential aspects of our different strategies. In particular, they allow us to tailor our behaviour based on past and future beliefs of the world. We now look at these two aspects in more detail. In the Robocup Rescue platform, damage (to civilians, rescue agents and buildings) is caused by fires. We believe that knowledge of how fires spread is essential to coordinate behaviours of agents. Thus, in our fire brigade strategy, we predict how fire spread. Such knowledge of potential damage of a particular fire site (that is spreading) would be useful when deciding which fire site warrants more extinguishing effort. Now, because of the uncertainties in the world, hidden simulation parameters such as temperatures of buildings have to be inferred. This often results in more inaccuracy in predictions the further we look in the future. Thus, it is usually better to predict only a few (10 in our fire brigade strategy) time steps ahead. Our agent runs its own simulation of the world with buildings burning and fires spreading (based on its current view of the world) for a number of time steps and uses its future belief of the world when deciding on which fire site is more critical.

5 Agent coordination and communication

As outlined in Section 2, the respective team centres make high-level decisions (at the strategic and tactical layers), which are then communicated to individual platoon agents. The latter use their own decision-making capabilities to carry out their assigned tasks (at the operational layer). As such, we take a mixed approach, where some decisions are centralised, while others are delegated to platoon agents. Apart from this, and in addition to the exchange of observations and information about the global world state (see Section 3), centres also communicate with each other to build up a more complete world state and to reason about their respective goals and plans. Such communication allows the different teams to coordinate their decisions where necessary. For example, if the current high-level goal of the ambulance team is to rescue a number of specific civilians, the police team can assist by clearing road blockages that are hindering the ambulances.

Finally, it is important to note that although our model relies on a number of centralised decision-makers, we are planning to incorporate various mechanisms for improving the robustness of the system. First, we believe that our current approach can cope with single agent failures relatively easily. As we assign tasks dynamically at the tactical layer, it is straight-forward to reassign the task of a failed agent to another one and to exclude it entirely from future task allocations (we could detect such failures when an agent persistently fails to report its observations). Now, should the centre itself fail, we can use an ordinary platoon agent to replace its role within the team. When all communication fails, we plan to incorporate simple single-agent strategies in each platoon agent to ensure that some sensible actions are still taken.

In the next section, we discuss the routing algorithm that our agents employ in order to efficiently navigate the map and predict travelling times.

6 Routing

Routing is an essential part of the operational decision-making layer of each agent. As mentioned before, the strategies for the ambulances, fire brigades and police forces critically depend on efficient computation of routes between two points on the map, as well as the availability of accurate estimated travel times. In our current implementation we use Dijkstra's algorithm to find the optimal route agents should follow to reach their destinations. The main issue with the use of Dijkstra's algorithm is the computational complexity, which is $O(E + V \log V)$ in the worst case, where V is the number of nodes of the graph and E the number of edges. In the RoboCup Rescue domain such a computational complexity can be problematic, because maps can be very big and the computation should be repeated every time a road changes its state (becomes blocked or is freed by a police agent). However, a good property of the Dijkstra's algorithm is that the output of its computation is a routing map from all nodes to a given source. Finding an optimal path from all nodes to the source, on this routing map, is then very easy. Therefore we can compute the routing map only once at each execution cycle considering as the source the starting position of the agent. To further enhance the computation speed, we also simplify the map (due to space restrictions, we do not cover this here).

This concludes our discussion of the basic decision-making procedures of our agents. In the following, we will outline in more detail some of the technical choices we have made in implementing our strategies.

7 Software Architecture

Our agent framework should be able to support a wide variety of strategies. It is therefore important that the software architecture of our individual agents should be as flexible as possible, without being too generic. To this end, we have redesigned the way in which agents determine their strategies and develop their plans, featuring a clear separation between the agent's beliefs, its strategic, tactical and operational aims, and the way in which it interacts with the outside world. One of the key aims of this redesigned architecture is ensuring modularity, so that individual components can be developed and improved with minimal impact on other components.

An overview of the architecture of the agent is shown in Figure 1. The agent state is modelled using a design pattern that closely resembles a blackboard. The state is updated by events from the environment, such as sensory input and messages received from other agents. Next, the strategic, tactical and operational reasoning modules are activated in turn. First, the strategic planner checks if new messages have been received from a central planner (i.e. a centre agent), and whether the conditions for the current strategic goal still hold. If not, it creates a new strategic goal and discards all lower level plans. In a similar way, the tactical reasoning subsequently checks the conditions for its current plans, concerning the prioritisation of tasks (such as rescuing civilians, extinguishing fires, and removing road blocks). Finally, the operational reasoning module determines how to best execute the current tactical plan. For example, this will involve finding a suitable route to the task location. The result of this reasoning is the execution of a particular action, and interacting with the communication module to exchange messages with other agents.

The blackboard pattern is very flexible, because it allows for the exchange of heterogeneous information between the planning modules. At the same time, the coupling between the reasoning modules is limited to the representation of the knowledge in the agent state. The software architecture presented in Figure 1 provides a high-level view of the operation of the agent. Finally, since the simulation framework has changed significantly since last year, we are in the process of changing the modules that are responsible for interfacing with the kernel.

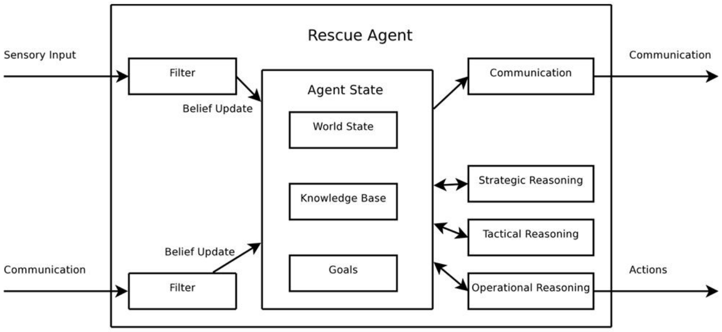

Fig. 1. Architecture of the Rescue Agent

8 Experience and Performance

In this section, we briefly discuss the history of our team and past performance. IAMRescue has participated in a number of Robocup tournaments, most recently in 2008. However, since then, our strategy has undergone major changes, including the novel police, ambulance and fire brigade strategies described in this report. The overall goal of our work is to apply principled and highly generic approaches to the Robocup Rescue domain, and so we strive to perform well in a wide range of settings — including those with no communication, unusual fire dynamics, dropped messages and missing agents or centres. As our software is currently being adapted to the new server, which was released this month, and due to space constraints, we decided not to include detailed performance logs in this report. Rather, we point the interested readers to a number of scientific papers that we have published on our strategies and which rigorously evaluate our techniques in varying settings [2, 3].

References

1. Siebra, C., Tate, A.: I-rescue: A coalition based system to support disaster relief operations. International Conference on Artificial Intelligence and Applications (AIA-2003) **101(1)** (2003) 289–294
2. Chapman, A., Micillo, R.A., Kota, R., Jennings, N.R.: Decentralised dynamic task allocation: A practical gametheoretic approach. In: The Eighth International Conference on Autonomous Agents and Multiagent Systems (AAMAS ’09). (2009) 915–922
3. Williamson, S.A., Gerding, E.H., Jennings, N.R.: Reward shaping for valuing communications during multi-agent coordination. In: The Eighth International Conference on Autonomous Agents and Multiagent Systems (AAMAS ’09). (2009) 641–648