BanzAI Team Description

Stefan Leijnen, Christian Vossers, Andrew Koster, and Silvain van Weers

Department of Computer Science, Utrecht University, The Netherlands
sp-is@cs.uu.nl
http://www.teambanzai.tk

Abstract. Using advanced pathfinding algorithms and a decision system based on utility-models for targets, we create agents that can largely act autonomously in a dynamic crisis scenario. We will show that using a clustering technique and the use of a new developed pathcost estimation algorithm the process of decisionmaking is simplified. The information given through communication and the organizational role of centers turn these agents into a well functioning team.

1 Introduction

The Robocup Rescue Simulation is a dynamic environment in which a group of heterogenous agents with different abilities have to operate optimally to complete their mission. The main purpose of the simulation is to save as many lives as possible in a limited amount of time, whereby the extinguishing of fires plays an important role. It forms an excellent situation for multi-agent cooperation and, because of the limited communication bandwidth, a framework for various communication techniques. Because of the three different types of agents we have analyzed different strategies for each of these types. We use dynamic pathfinding and utility-models to come up with a method that can be easily adjusted to any situation.

2 Decisionmaking

An agent scenario is divided into three considerations:

- An agent has no target
- An agent has one target
- An agent has more than one target

In the case of our first consideration an agent has to patrol to find targets. When an agent has only one target, a *priority* will be calculated for this target, but since there is only one target, the agent will try to move to this target in any case.
For agents that have more than one target the whole priority issue is very important. For multiple targets priorities are calculated for an agent, and eventually the agent will move to the target with the highest priority. The priority definition described here is a simplified version, where the priority depends on two main factors:

– Utility
– Reachability

Utility is measured by one or more properties of a certain target (e.g. fieryness and trapped civilians for burning buildings). For that reason the utility factor is target-specific. Reachability is a factor which an agent evaluates by considering the cost of moving to the target. The simplified equation for the priority of a target T for an agent a thus becomes:

$$Priority_a(T) = \frac{Utility(T)}{Reachability_a(T)}$$

2.1 Computing Pathcosts

We will briefly discuss the evaluation of the pathcost factor. This factor is defined by means of the value of the estimated pathcost to the target. The way this estimation is done depends on the (straight line) distance d between the agent and the target to be evaluated. Three cases are distinguished:

$d < \rho_1$ Dijkstra's shortest path algorithm is used to calculate the reachability.
$\rho_1 < d < \rho_2$ The $A*$ algorithm computes reachability, using the straight line distance as a heuristic.
$d > \rho_2$ A new algorithm, called $V*$, will be used to estimate reachability.

where ρ_1, ρ_2 are variables that can be modified to change the computational tractability of the decisionmaking procedure.
The $V*$ algorithm describes a method to estimate the value that the $A*$ algorithm would find on a pathfinding problem, using vectors as a heuristic. It simulates a function which, when given the vector between the agent and the target and vectors between the agent and certain other important objects (e.g. roadblocks), estimates the reachability. This function can be constructed as a lineair combination of these vectors, but it would be more efficient to let a neural network represent the function. Training this network can be done by reinforcement learning, using $A*$ to construct a training set.
We use this combination of algorithms for the sake of efficiency. Since Dijkstra's algorithm is too time consuming when applied to the entire map, the computationally less complex combination of $A*$ and $V*$ is used. By adjusting ρ_1 and ρ_2 we can finetune the complexity of the agents' calculations.

2.2 More on Utilities Concerning Specific Agents

Here we will describe different utility-models for each specific agent.

FireAgent. For modelling the problems for fireagents we described multiple fire ignitionpoints as `clusters`. Thus a cluster is a set, consisting of targets (burning buildings) and for that cluster a special utility is calculated. This utility could be, for instance, the average utility of all targets within that cluster. The decisionmaking for a fireagent is now simplified to making decisions about clusters instead of making decisions about all targets. Therefore the pathcost calculation becomes a lot more efficient.

PoliceForceAgent. Utilities for these agents are represented by road properties. Targets are blocked roads and utilities are calculated according to specific properties of a blocked road. For instance, a road with many lanes has a higher utility than a road with a small number of lanes. Therefore it evidently becomes more important for a policeagent to clear the roads with higher 'bandwidth' first, before clearing small roads.

AmbulanceAgent. Ambulances determine utilities on spots where civilians need help. This can be a building or just a normal (or blocked) road.

3 Communication

In this section, the centers come into play. First of all, they can distribute information on fires, roadblocks or (injured) civilians to agents, storing this in their memory so they will be able to make more educated decisions. The message sent contains information on both the object's properties as well as its location. Also, centers can order specific agents to move to a target by sending a request. For instance, when a fireagent becomes trapped between two roadblocks it can send out a request to it's center, which is passed on to the policecenter, that determines which policeagent is most suitable for clearing the roadblocks. This policeagent then receives a message that informs it about the roadblock's position on the map and it's utility. The latter is equal to the priority value of the fireagent's main target, which is usually high enough to convince the policeagent to head for these roadblocks. Consisting only of a location and a utility, multiple requests can be put into one message, increasing communication efficiency.

To give an overview of the solution to the dispatch problem, we imagine two main fire ignitionpoints in the disaster space. Suppose one ignitionpoint is a point from which fire spreads and creates maximum damage to both humans and infrastructure. For each fireagent this point automatically becomes the most important point to move to. The center oversees the situation and it is the center's decision to determine a different distribution of agents. When the center decides to send other agents to the other firepoint to which no agents were allocated initially. This can be achieved by making the utility of targets higher or lower according to how many known agents are already working at the problem sites. The center plays a crucial coordinating role and overrules any autonome behaviour of the agents itself temporarily.

4 Conclusion

BanzAI carries out the rescue operations with dynamic planning and dynamic decisionmaking. Clustering allows for simplification of the decisionmaking process and $V*$ creates a more efficient way of estimating the pathcost for certain targets. The utility-model and the communication (which uses this model) together solve the agent dispatch problem in an efficient way.

References

1. Satoshi Tadokoro, Hiroaki Kitano,"RoboCup Rescue", Kyoritsu Publisher 2000
2. S. Leijnen, A. Koster, C. Vossers, S. van Weers, "An Advanced Pathfinding Algorithm", `http://www.teambanzai.tk`, 2003