Contraction hierarchy algorithm that considers fault incidences under disaster environment NAITO-Rescue 2016 (Japan)

Kazuo Takayanagi[1], Shunki Takami[1], Yoki Miyamoto[1], Masahiro Yamamoto[1], Yoshiyuki Kozuka[1], Nobuhiro Ito[1], Kazunori Iwata[2]
rescue-2016@maslab.aitech.ac.jp

[1] Department of Information Science, Aichi Institute of Technology, Aichi, Japan
[2] Department of Business Administration, Aichi University, Aichi, Japan

Abstract. Various disaster recovery problems are presented as part of the RoboCupRescue Simulation (RCRS) project. We focus on the route search problem among these. To solve this problem, we propose a contraction hierarchy algorithm that considers faults. We describe agent implementations using our proposed method.

Keywords: Path Planning, Contraction Hierarchy, Disaster Environment

1 Introduction

The RoboCupRescue Simulation (RCRS) project aims to develop efficient agents that perform various disaster relief tasks. To perform efficient disaster relief, a key research problem that must be solved within the RCRS project is the shortest path search problem. In the RCRS project, agents move to targets to perform specific relief efforts. Hence, agents must find the shortest path to the targets in order to perform their activities smoothly and efficiently. Agents must be able to search and find such paths within short calculation times. This paper proposes a new method for solving the route search problem and describes the agent design using the proposed method. The rest of the paper is organized as follows. In Chapter 2, to resolve path search within the RCRS, we describe an adaptation of the contraction hierarchy algorithm. In Chapter 3, we describe agent implementations. Finally, in Chapter 4, we provide conclusions and avenues for future work.

2 Main Issues for NAITO-Rescue

In this chapter, to resolve path search within the RCRS, we describe an adaptation of the contraction hierarchy algorithm. We first provide an outline of this algorithm, including key definitions and the algorithm's importance. Next, we explain our proposed method, and then, report on our experiments and evaluation.

2.1 Contraction Hierarchy Algorithm

The contraction hierarchy algorithm, developed by Geisberger et al. in 2008, is divided into two stages : a precomputation stage and the shortest path calculation stage[1]. The precomputation stage prepares a hierarchy that stores the

nodes. The order of these nodes is determined by their relative importance. In this paper, we use edge difference as the method for computing importance. More specifically, we calculate the importance of each node given in the first graph, which we hereinafter call the input graph. The lowest node of importance is stored in the hierarchy. In this case, the hierarchy and the node directly corresponded to one another. When all nodes are in the hierarchy, the precomputation stage terminates. The shortest path search stage calculates the shortest path based on the precomputation. From these results, we obtain an actual path based on the graph calculated in the precomputation stage. Using this approach, the contraction hierarchy algorithm performs the shortest path search.

2.2 Key Definitions

The key definitions are shown in Table.1 and 2.

Table 1. Definitions of symbols

Key	Explanation
G	The weighted graph
V	The set of nodes
E	The edge connecting two nodes
C	A cost function
B	A fault incidence function

Table 2. Definitions of terms

Name	Definition
Graph	We denote a road network as a weighted graph G, which is defined as $G = (V, E, C)$
Path	A path from v_1 to v_k is denoted as a sequence of nodes $P = (v_1, v_2, \ldots, v_k) \in V \times V \times \cdots \times V$ with v_i adjacent and connected to v_{i+1} by $e_{i,i+1} \in E$ for $1 \leq i < k$
Cost of Path	The cost of path $(v_1, v_2, ..., v_k)$ is as follows: $c_{1,2} + c_{2,3} + c_{3,4} + ... + c_{(k-1),k} (2 \leq k \leq n)$
Shortest Path	The shortest path from v_s to v_g is path $P = (v_1, v_2, \ldots, v_n)$ ($with\ v_1 = v_s\ and\ v_n = v_g$) that, over all possible n, minimizes $\sum_{i=1}^{n-1} c_{i,i+1}$; the cost of the shortest path is called *a minimum cost for the path*
Shortcut	New edges $e_{x,y}$ that represent the shortest path $(v_x, ..., v_y)$
Agents	An agent moves similarly to a vehicle from a start node to a destination node using a map.
Fault Information	Fault information denotes information about blocked roads that an agent cannot pass through.
Detour Path	The next-best shortest path when there is a fault on the shortest path.

2.3 Importance

As mentioned earlier, measuring importance uses the edge difference. The value of the edge difference of node v is the difference between the number of edges

connected to node v and the number of shortcuts belonging to the same level as the node. In Fig. 1, we show an illustrative example of edge difference. In the figure, solid lines show edges, while dotted lines show shortcuts that are at the same level as the given node. Further, the numbers on each edge or shortcut represents cost.

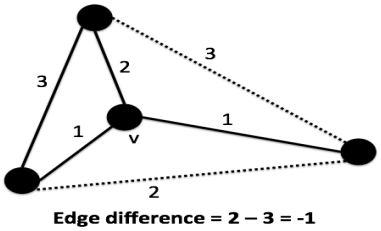

Fig. 1. Illustrating the edge difference

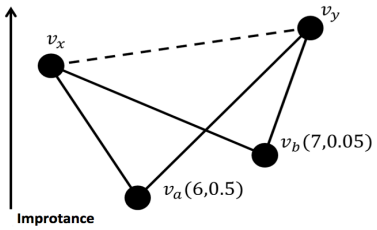

Fig. 2. Shortcut and detour paths

2.4 Contraction Hierarchy Algorithm that Considers Fault Incidences

We introduced fault incidence into the contraction hierarchy algorithm; thus, in this section, we explain our extended contraction hierarchy algorithm. Specifically, there are two paths $((v_x, v_a, v_y), (v_x, v_b, v_y))$, which together become a shortcut, as shown in Fig. 2. In node $v(m, n)$ of Fig. 2 , m represents cost and n represents a fault incidence. Further, if three or more paths exist, we randomly choose two paths; the other path becomes the preparatory path.

In the RCRS, when a failure occurs along the shortest path, we must then consider a detour path. Here, we choose a path in which the predicted distance considers the detour path as the shortest. We calculate cost $c_{x,y}$of the shortcut that connects v_x and v_y.

We obtain the predicted moving distance considering a fault incidence to determine $c_{x,y}$. Moreover, it is necessary to compare the cost of the resulting path. In Fig. 2, when path α and path β are set to (v_x, v_a, v_y) and (v_x, v_b, v_y), respectively costs defined as c_α and c_β. Also, fault incidences of edges $e_{x,a}$ and $e_{a,y}$ are denoted $b_{x,a}$ and $b_{a,y}$. Therefore, when fault incidence b_α occurs on path α, b_α is defined as shown below. Similarly, we define fault incidence b_β for path β.

$$b_\alpha = 1 - (1 - b_{x,a})(1 - b_{a,y})$$

The shortcut is composed of two or more edges; however, we want to perform a calculation while maintaining the shortcut, which is a feature of the contraction hierarchy algorithm, wherever possible. Hence, we perform the calculation without deploying the shortcut. Therefore, it is necessary to calculate the cost and fault incidence of the edge from the cost and fault incidence saved as a shortcut. In this case, costs and fault incidences of the two edges are approximately calculated as an equivalence. Therefore, costs $c_{x,a}$, $c_{a,y}$ and fault incidences $b_{x,a}$, $b_{a,y}$ of edges $e_{x,a}$, $e_{a,y}$ of path α are defined below. Costs $c_{x,b}$, $c_{b,y}$ and fault incidences $b_{x,b}$, $b_{b,y}$ of edges $e_{x,b}$, $e_{b,y}$ of path β are calculated similarly.

$$c_{x,a} = c_{a,y} = \frac{c_\alpha}{2}, b_{x,a} = b_{a,y} = 1 - \sqrt{1 - b_\alpha}$$

By using the above, we propose the following equation to calculate the predicted moving distance $s_\alpha(x,y)$, $s_\beta(x,y)$ of path α, β in consideration of a fault incidence from x to y.

$$s_\alpha(x,y)=\frac{1}{1-b_\alpha}\{c_\alpha+(1-b_\alpha)c_\alpha+(1-\sqrt{1-b_\alpha})c_\beta+\sqrt{1-b_\alpha}(1-\sqrt{1-b_\alpha})(c_\beta+\frac{c_\alpha}{2})\} \quad (1)$$

Here, if the fault incidence is low, the value of 1 increases. Note that $(1-b_\alpha)c_\alpha$ is the expected value for the case in which path α can pass. Further, $(1-\sqrt{1-b_\alpha})c_\beta$ is the expected value for the case that traverses path β when a fault occurs in edge $e_{x,a}$. Finally, $\sqrt{1-b_\alpha}(1-\sqrt{1-b_\alpha})(c_\beta+\frac{c_\alpha}{2})$ is the expected value for passing through path β and turning back when a fault occurs in edge $e_{a,y}$.

In the extended algorithm, the optimal path is selected as the smaller one by comparing the predicted moving distance values calculated in (1). If three or more paths exist, the optimal path selects the value of the smaller one. Next, we randomly select a path from among the spare paths. Therefore, we compare the optimum route and selected path values calculated by (1). This calculation continues until the spare paths are gone.

2.5 Our Proposed Method

The process flow of our proposed method is as follows.

Precomputation

1. Calculate the importance of the given node, and then add it to the hierarchy.
2. Remove the lowest-valued importance node from the hierarchy.
3. Add the shortcut to the input graph.
4. If the shortcut that connects the pair of nodes of the added shortcut already exists, leave the one for which the calculation results of (1) are smaller.
5. Repeat steps 1 through 4 until the number of nodes that belong to the hierarchy becomes two.

Shortest Path Search

1. Generate a list of paths to the nodes connected to the starting point (hereinafter the start side list).
2. Generate a list of paths to the node connected to the destination(hereinafter the destination side list).
3. Calculate the cost of nodes connected by edges that contains a shortcut from the start point in (1); search this path.
4. If the path that connects to the destination is found, output the path; however, if the path is the shortcut, expand it to the path, output the path, and exit.
5. Add the path to the start side list. This path is the path to a node connected by an edge that contains the shortcut to the selected node from the start point side; however , this path does not contain the start point.
6. Repeat, as much as possible, steps 3 through 6, during which if the same path containing a node of the search target in the start side list in step 3 exists, the path of that node is selected by using the smaller value calculated by (1).

7. Calculate the cost of nodes connected by edges that contains a shortcut from the destination in (1); search this path.
8. If the path that connects to the start point is found, output the path; however, if the path is the shortcut, expand it to the path, output the path, and exit.
9. Add the path to the destination side list. This path is the path to a node that is connected by an edge that contains a shortcut to the selected node from the destination side; however, this path does not contain the start point.
10. If there is a node of the search target that leads to the last point of the path of the start side list, the path that added the destination side list to the start side list becomes a candidate for the optimum route.
11. Repeat, as much as possible, steps 7 through 10, during which if the same path of a node of the search target of the start side list in step 7 exists, the path of that node is selected based on the smaller value calculated by (1).
12. If two candidate paths exist for the optimal path, the optimal path is selected based on the smaller value calculated by (1).
13. If it contains shortcuts in the optimal path, expand it to path.
14. Output the path and exit.

2.6 Evaluation Experiments

Experimental Methodology

We compared our proposed method to Dijkstra 's algorithm [2] and the original contraction hierarchy algorithm from the following viewpoints:

- Moving distance
- The number of faults discovered
- The number of nodes the search process passes through (hereinafter the number of hops).

The precomputation stage of our proposed method and that of the contraction hierarchy algorithm, as well as the map data and its fault incidence composed via a graph, were all given in advance. Further, because in our simulation, we assume our experiments are aimed at the environment after a disaster, we consider it appropriate to perform the precomputations before the disaster. The moving distance and the number of faults discovered versus, the number of hops yield a ratio of results for Dijkstra's algorithm for the case in which failure dose not occur. Further, the number of faults discovered was the number of faults encountered per 100m.

Maps for Experiments

We used 1/25,000-scale map data for all areas in Japan. These maps were released by the Geographical Survey Institute [3] and the data were expressed in G-XML, which is a Japanese Industrial Standards format. We used the information regarding roads, which consists of nodes for roads and road edges. Nodes for roads depict intersections V in weighted graph G Road edges are roads E in weighted graph G, each of which connects the interval between two road nodes. Cost function C draws its length from each edge. Blocked road rate function B was invented by the liquefaction hazard map of cities [4]. We used two areas to perform our experiments, as summarized in Table 3, and performed 10,000 experiments for each area. In each experiment, the start and destination nodes of an agent were randomly selected.

Table 3. Information regarding the two sample areas

Area No.	Area	$\|V\|$	$\|E\|$
1	Atsuta ward	1,760	2,609
2	Minato ward	4,892	7,127

Table 4. Results of our experiments

	The moving distance		The number of faults discovered	The number of hops	
	Avg.	S.D.	count	Avg.	S.D.
Area 1: Atsuta ward					
Dijkstra's	1.66080	4.85629	0.20313	1.5489	0.6884
Contraction hierarchy	1.50747	4.68011	0.21674	1.5501	0.6899
Our proposed method	1.36081	0.79169	0.03127	1.3369	0.7360
Area 2: Minato ward					
Dijkstra's	1.45009	1.03418	0.26991	1.5127	0.5092
Contraction hierarchy	1.39170	0.39416	0.27670	1.5131	0.5091
Our proposed method	1.26117	0.33277	0.02286	1.1587	0.3493

Results and Discussion

In our proposed method, we observed that the moving distance, the number of encountered faults, and the number of hops decreased as compared to the other existing methods in an environment where faults occurred. Calculation results of the existing methods indicated that there are edges with high fault incidences, because such methods do not consider fault incidence of edges. Our proposed method calculates a path in consideration of avoiding edges with high fault incidences, i.e., calculation results are difficult for existing edges with high fault incidences. Therefore, we conclude that using our proposed method, it is possible to calculate a more efficient path.

3 Agent Implementations

In this section, we describe the agent behavior used in the algorithm proposed in Chapter 2.

3.1 Agent Design

Our team developed agent implementations using the Agent Development Framework (ADF) [5]. We implemented our proposed methods for the algorithm modules using ADF. Detailed implementations were written for each agent section.

3.2 FireBrigade

FireBrigade agents play the role of extinguishing fire. The purpose here is to minimize the spread of fire. In this section, we describe the decisions made regarding the workspace and target selections for FireBrigade.

First, we describe our decisions regarding workspace. We allocate a workspace to each FireBrigade to arrange agents with workloads and reduce the amount of required calculations. To allocate a workspace to FireBrigade, we use the k-means clustering algorithm. The process of the adopted k-means [6] algorithm is as follows.

1. Randomly assign a cluster to each building xi.
2. Based on the assigned buildings, determine center Vj of each cluster in the calculation.
3. Obtain the distance between xi and Vi, assigning xi to the nearest cluster.
4. Repeat steps 2 and 3 until there is no change in the center.

Next, we describe how target selection works. We define fire clusters as groups of burned buildings. We evaluate the distance between the center of a fire cluster and the center of the map, classifying fire clusters into two cases based the distance.

Case 1: Near the center of the map

Here, the distance is short meaning that the fire cluster is close to the center of the map. It is therefore very likely to spread to exteriors, thus extinguishing from the most exterior in range is the approach taken here. If there are more than one identified as the most exterior, the building with the highest temperature is selected.

Case 2: Far from the center of the map

Here, the distance is long, meaning that the fire cluster is far from the center of map. The damage caused by the spread of fire is comparatively minor. Therefore, extinguishing the fire from the central building is the approach taken.

3.3 PoliceForce

PoliceForce agents play the role of clearing blockades. The purpose here is to enable other agents to be able to move to destinations via the shortest path. In this section, we describe our decisions regarding workspaces and how to clear blockades.

First, we describe our decisions regarding workspace. To allocate a workspace to PoliceForce, we use the k-means algorithm, just as with the FireBrigade. PoliceForce act as follows in the cluster.

1. Clear blockades from the allocated cluster to refuge.
2. Clear blockades in the cluster in the order of priority evaluated by our contraction hierarchy algorithm.
3. Clear remaining blockades in the cluster and gather new information.

Next, we describe how to clear blockades. PoliceForce agents clear a blockade by using coordinates and vectors. To efficiently clear blockades, we think it is effective to clear the center blockades of the road. In this case, PoliceForce agents partition the road into triangles and calculate its center. In a triangle, the center of a side that does not correspond with a road edge and the center of an edge that is contiguous with a road or building as coordinates act to clear.

3.4 AmbulanceTeam

AmbulanceTeam agents play the role of rescuing civilians. The purpose here is to find and rescue buried civilians to the extent possible and carry them to refuge. In this section, we describe our decisions regarding workspaces and target selections of AmbulanceTeam.

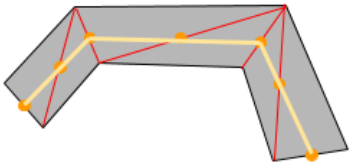

Fig. 3. The blockade removal path

First, we describe our decisions regarding workspaces. To allocate a workspace to AmubulanceTeam, we use the k-means algorithm, just as with the other agents. AmubulanceTeam agents rescue civilians and gather information in the cluster.

Next, we describe target selection. AmubulanceTeam agents have civilian information sent from other agents and acquired by the agents themselves. AmubulanceTeam agents select a civilian to rescue from this information. In the selection of a civilian, the agents judge whether they will be able to rescue the said civilian given the distance between the AmubulanceTeam and the civilian as well as the damage the civilian has experienced. If there are civilians with slight damage, they are carried to a road temporarily while others are being rescued. After rescuing all civilians, they are carried to refuge.

4 Conclusions

In this paper, we focus on the path planning problem and propose a new method for solving this problem. Next, we describe new strategies for agents. In the future, we plan to implement our algorithm described in this TDP. If possible, we also plan to compare our approach with other algorithms and implement our algorithm more effectively.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Numbers JP 16K00310 and 26330166.

References

1. Geisberger, R., Sanders, P., Schultes, D., & Delling, D. 2008. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. Experimental Algorithms, 319-333.
2. Dijkstra, E.W. 1959. A Note on Two Problems in Connexion with Graphs. Numerische Mathematik 1, 269-271.
3. Geospatial Information Authority of Japan. 2003. Gsi:1/25,000 map information of japan. http://www.gsi. go.jp (in Japanese).
4. Nagoya City Hall. 2004. Earthquake map of nagoya city (in japanese). http://www.city.nagoya.jp/kurashi/category/20-2-5-6-0-0-0-0-0-0.html
5. Kazuo Takayanagi, Shunki Takami, Nobuhiro Ito, Kazunori Iwata. RCRS-ADF. https://github.com/RCRS-ADF/RCRS-ADF
6. MacQueen, J. B. 1967. Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. 281-297