Real Time Critical Edge of the Shortest Path in Transportation Networks*

Yinfeng Xu[1,2] and Huahai Yan[1]

[1] School of Management, Xi'an Jiaotong University, Xi'an, 710049, P.R. China
[2] The State Key Lab for Manufacturing Systems Engineering, Xi'an, 710049, P.R. China.
{yfxu, yanhai}@mail.xjtu.edu.cn

Abstract. In transportation networks, a vehicle always travels longer than the shortest path due to sudden edge failure caused by unexpected events such as accident. In this situation, which edge failure results in the maximum of the travel distance between the source node and the destination node? If we know the edge, we can reduce the transportation cost and improve the networks structure. Regarding this problem, the most vital edge (MVE) problem considers in a global view and from the perspective of static decision-making based on complete information, while the longest detour (LD) problem solves in a local view and in terms of real time. This paper reconsiders this problem in a global view and in terms of real time. We propose the real time critical edge (RTCE) problem of the shortest path, and present an $O(n^2)$ time algorithm by constructing the shortest path tree. Then, by giving a numerical example of urban transportation networks, we compare the results of MVE, LD and RTCE, and conclude that the RTCE problem has more practical significance.

Keywords: Real Time Critical Edge, The Shortest Path, Algorithm, Transportation Networks.

1 Introduction

In urban transportation, there are always many road blockages caused by unexpected events such as accidents. These sudden events make the vehicles to detour thus lengthening the whole travel distance and increasing the transportation cost. In fact, these events are unforeseen, particularly, one can not obtain complete information regarding the blockages during the process of travelling. It is important to know the real time critical edge of the shortest path for transportation management. Studying the real time critical edge of the shortest path provides scientific basis for raising transportation efficiency and reducing the loss caused by the real time critical edge failure.

* This research is supported by NSF of China under Grants 70525004, 10371094 and 70471035.

J.-Y. Cai, S.B. Cooper, and A. Li (Eds.): TAMC 2006, LNCS 3959, pp. 198–205, 2006.

Previous related research mainly focused on the most vital edge (MVE) problem and the longest detour (LD) problem. The MVE problem was originally presented by Corley and Sha [1] who studied the problem of finding an edge whose removal from the graph $G(V, E)$ results in the largest increase of the distance between two given nodes. This edge is generally denoted as the most vital edge with respect to the shortest path. This problem has been solved efficiently by K. Malik, A. K. Mittal and S. K. Gupta [2], who gave an $O(m + n \log n)$ time algorithm by making use of Fibonacci heap. E. Nardelli, G. Proietti and P. Widmyer [3] improved the previous time bound to $O(m \cdot \alpha(m, n))$, where α is the functional inverse of the Ackermann function, and n and m denote the number of nodes and edges in the graph, respectively.

The LD problem was first introduced by E. Nardelli, G. Proiett and P. Widmyer [4]. They focused on the problem of finding an edge $e = (u, v)$ in the shortest path where u is closer to source node s than v , such that when this edge is removed, the distance of detour satisfies $d_{G-e^*}(u^*, t) - d_G(u^*, t) \geq d_{G-e}(u, t) - d_G(u, t)$, where $G - e = (V, E - e)$, $e^* = (u^*, v^*)$. This problem was denoted as the *longest detour (LD) problem*, and the edge whose removal will result in the longest detour is named the *detour-critical edge*. They showed that this problem can be solved in $O(m + n \log n)$ time, and then [3] improved the result to $O(m \cdot \alpha(m, n))$ time bound. In addition, there are some other related literatures focusing on this problem such as E. Nardelli, G. Proiett and P. Widmyer [5], LI Yinzhen and GUO Yaohuang [6], and A. M. Bhosle [7].

In the MVE problem, decision-making of route is made based on the complete information, namely the decision maker knows in advance which edge is destroyed. In this sense, the MVE problem does not consider the real time situation under incomplete information. In addition, the LD problem merely focuses on the distance of detour in a local view thus neglecting the distance before detour. Aiming at improving the deficiency mentioned above, this paper presents the RTCE problem which contributes (1) Considering the problem from the view of real time under incomplete information and (2) Computing the whole route in a global view including not only the distance of detour but also the distance before detour.

This paper is organized as follows: in section 2 we give definition of the real time critical edge of the shortest path; in section 3 we present an algorithm to solve the RTCE problem efficiently and analyze its time complexity; in section 4, we show a numerical example of urban transportation networks to illustrate the application of the algorithm; finally, section 5 contains concluding remarks and lists some further research problems.

2 Problem Statement and Definition

Model a realistic road transportation networks as a graph in which the intersection could be abstracted as the node and the road between two nodes as the edge. The weight of edge could be represented as the distance of the road. For the sake of brevity, we will refer to road transportation networks as transportation networks and denoted as $G(V, E)$. Let $V = \{s, v_1, v_2, \ldots, v_{n-2}, t\}$ denotes the set of nodes and $E = \{e_1, e_2, \ldots, e_m\}$ denotes the set of edges, where n and m denote the number of nodes and edges in the graph respectively.

All discussions in this paper are based on the following assumptions:

1. The vehicle always travels along the shortest path from source node s to destination node t. If an edge $e = (u, v)$ that is on the shortest path fails, the vehicle will travel along the shortest path (from u to t) that does not use edge $e = (u, v)$, where node u is nearer to source node s than node v.
2. Sudden failure of edge only happens on the shortest path from source node to destination node, and we suppose that the shortest path between two nodes is unique in this paper.
3. During the process of travelling, there is only one edge sudden failure.
4. The information of "edge failure" could be obtained when the vehicle travels to node u of the failure edge (u, v), in particular, node u is nearer to source node s than node v .

The MVE problem is based on the static situation under complete information. Here, considering the real time situation in which information of edge failure is incomplete, we define the real time critical edge (RTCE) problem of the shortest path which consider real time situation and includes both the distance of detour and the distance before detour.

Definition. In an 2-edge connected, undirected graph $G(V, E)$. Given source node s and destination node t, the shortest path $P_G(s, t)$ from s to t in G is defined as the shortest path which minimizes the sum of the weights of the edges along $P_G(s, t)$. A detour at node $u \in P_G(s, t) = \{s, \ldots, u, v, \ldots, t\}$ is defined as the shortest path from u to t which does not make use of edge $e = (u, v) \in P_G(s, t)$ with u is closer to s than v, and let $d_{G-e}(u, t)$ denotes the travel distance of detour from u to t in $G(V, E - e)$. Here we focus on the problem of finding an edge $e^* = (u^*, v^*) \in P_G(s, t)$ whose removal results in the longest travel distance $d_{G-e^*}(s, t) \geq d_{G-e}(s, t)$ for every edge of $P_G(s, t)$, where $G - e = (V, E - e)$. For the sake of brevity, we will refer to this problem in the following as the *real time critical edge* (*RTCE*) problem of the shortest path, where $d_{G-e^*}(s, t) = d_G(s, u^*) + d_{G-e^*}(u^*, t)$, $d_{G-e}(s, t) = d_G(s, u) + d_{G-e}(u, t)$, and $d_G(s, u)$ is denoted as the travel distance between s and u before detour.

In realistic situation, one can not obtain the complete information before setting out because of sudden accidents. Therefore, research on the RTCE problem has extremely practical significance.

3 Solving the RTCE Problem

Let $P_G(s, t)$ be the shortest path joining s and t in G. It is worth noting that solving the RTCE problem in the naive way that is by sequentially removing all the edges $e = (u, v)$ along $P_G(s, t)$ and computing at each step $P_{G-e}(s, t)$. In fact, this leads to a total amount of time of $O(n^3)$ for the $O(n)$ edges in $P_G(s, t)$. Especially, if there is large number of nodes in the networks, the computation will cost too much time.

We now discuss an improved approach. Since that the distance before detour does not change after deleting the failure edge, we need only compute the distance of detour for any edge along the shortest path $P_G(s,t)$, thus avoiding repeatedly computing the distance before detour at each step. As we know, the whole route from s to t includes the distance before detour and the distance of detour. Note that the distance before detour must be on $P_G(s,t)$ and could be obtained directly which denoted as $d_G(s,u)$, so we need only compute the shortest path joining node u and destination node t thus reducing the complexity, where node $u \in e = (u,v)$ is nearer to source node s than node v.

In what follows, we present this approach in detail. We start by computing the shortest path tree rooted at t denoted as $S_G(t)$. This gives us all the shortest paths toward destination node t, where we suppose that the shortest path is unique. Let $e = (u,v)$ be an edge on $P_G(s,t)$ with u closer to s than v. When edge e is removed, the shortest path tree $S_G(t)$ is partitioned into two subtrees, as shown in Figure 1. Let $M_t(u)$ denotes the set of nodes reachable in $S_G(t)$ from t without passing through edge $e = (u,v)$ and let $N_t(u) = V - M_t(u)$ be the remaining nodes, then $N_t(u)$ could be regarded as the shortest path tree rooted at u (i.e., the subtree of $S_G(t)$ rooted at u). Note that for the nodes in $M_t(u)$, the distance from t does not change after deleting edge e, while for the nodes in $N_t(u)$ the distance from t may increase as a consequence of deleting edge e.

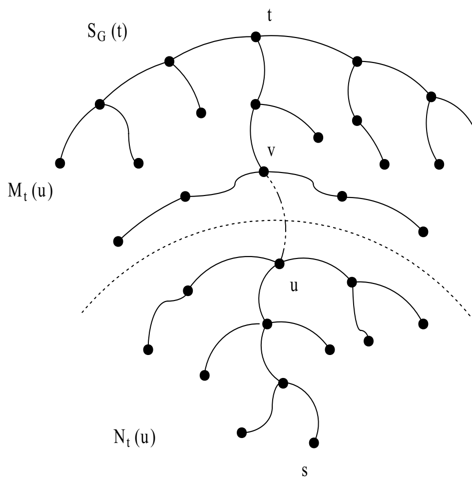

Fig. 1. Edge (u,v) is removed, the shortest path tree $S_G(t)$ is partitioned into $M_t(u)$ and $N_t(u)$

According to the analysis above, we can yield the shortest path tree rooted at u and t, respectively. Figure 1 illustrates this situation. Since the detour joining u and t must contain an edge as the linking edge $w(x,y)$, in particular, $x \in N_t(u)$, $y \in M_t(u)$, it follows that it corresponds to the set of edges whose weights satisfy the following condition, figure 2 illustrates this situation.

$$d_{G-e}(u,t) = \min_{x \in N_t(u), y \in M_t(u)} \{d_{G-e}(u,x) + w(x,y) + d_{G-e}(y,t)\}$$

In fact, $x \in N_t(u)$, such that

$$d_{G-e}(u,x) = d_G(u,x) = d_G(t,x) - d_G(t,u)$$

Also, since $y \in M_t(u)$, so

$$d_{G-e}(y,t) = d_G(y,t)$$

Therefore

$$\begin{aligned} d_{G-e}(u,t) &= \min_{x \in N_t(u), y \in M_t(u)} \{d_{G-e}(u,x) + w(x,y) + d_{G-e}(y,t)\} \\ &= \min_{x \in N_t(u), y \in M_t(u)} \{d_G(t,x) - d_G(t,u) + w(x,y) + d_G(y,t)\} \end{aligned}$$

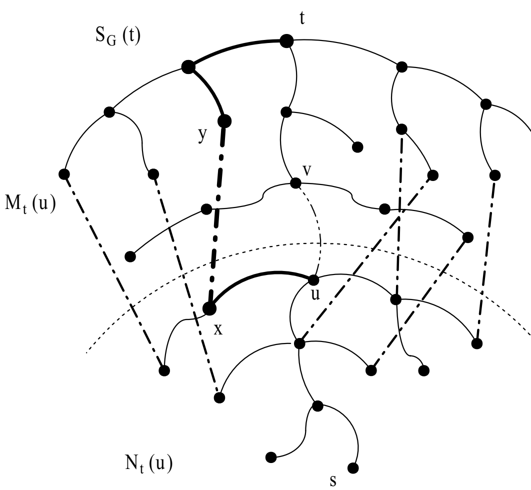

Fig. 2. Edge (u, v) is removed, dashed lines represent the linking edges. In bold, the detour at u with its linking edge (x, y).

Now, add the distance before detour $d_G(s,u)$ which has already been obtained, we can compute the whole travel distance as follow

$$d_{G-e}(s,t) = d_G(s,u) + d_{G-e}(u,t)$$

3.1 Algorithm

The following algorithm for obtaining the real time critical edge is based on the results mentioned above.

Step 1: Compute the shortest path from t to all the other vertexes by using the algorithm of *Bellman-Ford*. Meanwhile, record $P_G(s,t)$, $S_G(t)$ and k (the number of edges along $P_G(s,t)$).
Step 2: Set $i = 1$
Step 3: Remove edge e_i from $P_G(s,t)$ thus produce $S_{G-e_i}(t), M_t(u), N_t(u)$
Step 4: Compute

$$d_{G-e_i}(s,t) = d_G(s,u) + \min_{x\in N_t(u), y\in M_t(u)} \{d_G(t,x) - d_G(t,u) + w(x,y) + d_G(y,t)\}$$

Step 5: Set $i = i+1$, if $i \leq k$, then go back to step 3. Otherwise, go to step 6
Step 6: Compute $d_{G-e^*}(s,t) = \max_{i=1,\ldots,k} \{d_{G-e_i}(s,t)\}$, the edge e^* which maximizes $d_{G-e_i}(s,t)$ is the real time critical edge.

3.2 Analysis of Algorithm Complexity

Now we discuss its time complexity for a planar transportation networks, where n and m denote the number of nodes and edges in $G(V,E)$.

1. In step 1, the set of the shortest path can be obtained by the algorithm of *Bellman-Ford* in $O(mn)$ time
2. In step 3, we obtain a total time of $O(1)$
3. The time for step 4 is $O(m)$
4. Step 2-5 are loop computation and its repeat times is k. Since $k \leq n-1$, the total time for step 2-5 is $O(mn)$
5. The time for step 6 is $O(n)$

It follows that the time complexity of this improving algorithm is $O(mn)$. Specifically, for a planar transportation networks, $m = O(n)$, so the total time complexity of the algorithm presented this paper is $O(n^2)$.

4 Numerical Result

We investigate part of urban transportation networks as illustrated in Figure 3. A transportation company sends a vehicle to carry the freight from source node s to railway station t. The vehicle travels along the shortest path from s to t, if the road to the station is blocked because of accident, the vehicle has to detour to the station. In this situation, in order to reach the railway station in time, when should the vehicle set out?

In fact, solving this problem means to find the real time critical edge in terms of real time. If we identify the real time critical edge of the shortest path, we can determine the longest travel distance from s to t when the real time critical edge failure. From the view of transportation management, it is a significant issue.

As illustrated in Figure 3, the shortest path $P_G(s,t)$ from source node s to destination node t is $s \rightarrow v_4 \rightarrow v_9 \rightarrow t$, and its distance is 32. We compute the MVE, LD and RTCE problem. The numerical results are as follows. See Table 1.

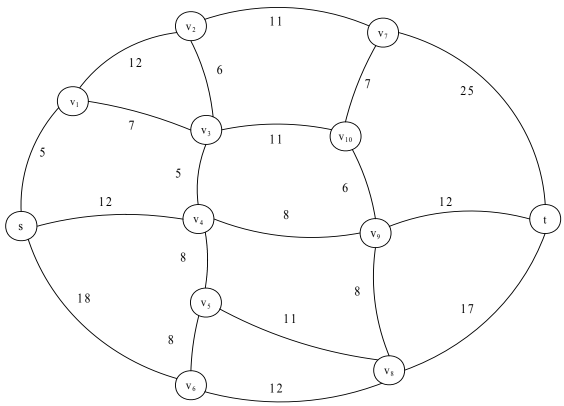

Fig. 3. Part of urban transportation networks

Table 1. Numerical Result

	MVE	LD	RTCE		
	$d_{G-e}(s,t)$	$d_{G-e}(u,t) - d_G(u,t)$	$d_G(s,u)$	$d_{G-e}(u,t)$	$d_{G-e}(s,t)$
(s, v_4) fails	37	$37 - 32 = 5$	0	37	37
(v_4, v_9) fails	41	$34 - 20 = 14$	12	34	46
(v_9, t) fails	45	$25 - 12 = 13$	20	25	45
e^*	$edge(v_9, t)$	$edge(v_4, v_9)$	$edge(v_4, v_9)$		

From the numerical result, we can make some comparisons among the three problems: the MVE problem, the RTCE problem and the LD problem.

Firstly, note that in the RTCE problem the vehicle travels one unit more than it does in the MVE problem. This is because the RTCE problem is a real time process under incomplete information, which implies the vehicle can not get the information of edge failure until it travels to the blockage edge, and the travelling route of the RTCE problem is $s \to v_4 \to v_3 \to v_{10} \to v_9 \to t$; But in the MVE problem the information could be totally obtained in advance, and the travelling route of the MVE problem is $s \to v_4 \to v_9 \to v_8 \to t$.

Then, let us see the difference between the RTCE problem and the LD problem. Obviously, the travelling route of the LD problem is $v_4 \to v_3 \to v_{10} \to v_9 \to t$ and its distance is 34; While the distance of the RTCE problem is 46. This is because the RTCE problem targets at the whole travel distance in a global view while the LD problem only considers the distance of detour in a local view.

Finally, the critical edge of the LD, MVE and RTCE problem are different. The critical edge of the LD and RTCE problem are edge (v_4, v_9), but the critical edge of the MVE problem is edge (v_9, t). Which is the critical edge depends on the network structure of the urban transportation networks given in this paper. If the structure changes, the result will be changed, which depends on the structure of the transportation networks.

In realistic transportation networks, those sudden edges failure are unforeseen, particularly, the vehicle does not get the information of edge failure until it travels to the failure edge. According to comparisons above, we can conclude that the RTCE problem has more practical significance, and from the view of transportation management, the RTCE problem which focuses on a real time process is an important problem and worthwhile to consider.

5 Conclusions

In urban transportation, there are always many road blockages caused by unexpected events such as accidents. Since these sudden events are unforeseen, making it more difficult to choose an optimal travel route, finding the real time critical edge of the shortest path under incomplete information has further practical significance for transportation management. This paper presents a detailed algorithm whose time complexity is $O(n^2)$ and gives a realistic case of urban transportation networks to illustrate the application of our algorithm. We compare the results of the MVE, LD and RTCE problem, and conclude that the RTCE problem has more practical significance. We can further reduce the time complexity of algorithm by making use of Fibonacci heap or the functional inverse of the Ackermann function. In addition, the real time critical node of the shortest path can be studied as future work.

References

1. H. W. Corley, D. Y. Sha. Most vital links and nodes in weighted networks. *Operation Research Letters*, **1**:157-161, 1982.
2. K. Malik, A. K. Mittal, S. K. Gupta. The k most vital arcs in the shortest path problem. *Operation Research Letters*, **8**:223-227, 1989.
3. E. Nardelli, G. Proietti, P. Widmyer. A faster computation of the most vital edge of a shortest path between two nodes. *Information Processing Letters*, **79**(2):81-85, 2001.
4. E. Nardelli, G. Proietti, P. Widmyer. Finding the detour critical edge of a shortest path between nodes. *Information Processing Letters*, **67**(1):51-54, 1998.
5. E. Nardelli, G. Proietti, P. Widmyer. Finding the most vital node of a shortest path. *Theoretical Computer Science*, **296**:167-177, 2003.
6. LI Yinzhen, GUO Yaohuang. Study on vital edges of shortest paths in traffic and transportation networks. *Chinese Journal of Management Science*, **12**(4):69-73, 2004.
7. Amit M. Bhosle. Improved algorithms for replacement paths problems in restricted graphs. *Operations Research Letters*, **33**:459-466, 2005.