KSCE Journal of Civil Engineering (2010) 14(1):41-49
DOI 10.1007/s12205-010-0041-8

*Transportation Engineering*
www.springer.com/12205

# A Dissimilar Alternative Paths-Search Algorithm for Navigation Services: A Heuristic Approach

**Yeon-Jeong Jeong*, Tschangho John Kim**, Chang-Ho Park***, and Dong-Kyu Kim******

Received April 27, 2009/Accepted July 7, 2009

## Abstract

The purpose of this paper is to present the development of a heuristic algorithm that searches dissimilar alternative paths efficiently for multi-route problems in navigation services. The K shortest paths algorithm, one of the algorithms that provides multiple paths, often provides very similar paths between an origin and a destination, oftentimes differentiated by only one different link. The algorithm developed in this paper is based on the renewed path-partition-deletion method, which searches for the shortest paths by executing the shortest-path algorithm only once. Users can obtain information on the alternative paths corresponding with their preferences by the user-specified allowable cost and duplication ratios included in the algorithm. Empirical studies based on a toy network and a Chicago network are presented to evaluate the efficiency of the developed algorithm. The results show the algorithm can be applied to the implementation of the paths-search algorithm for multi-route navigation services.

Keywords: *dissimilar alternative path, navigation service, K shortest paths algorithm, path-partition-deletion method*

## 1. Introduction

According to a report by Berg Insight, 16.5 million units and 11 million units of Personal Navigation Devices (PNDs) were sold in Europe and North America, respectively, in 2007. Sales in Europe and North America are forecasted to grow to 26 million and 27 million units, respectively, by 2014. A survey by LG's *Economic Weekly* showed that 1.2 million PNDs were sold in Korea in 2006. In addition, it further forecasted that the market for Location-Based Services (LBS) in Korea would reach about $110 million in 2008.

The crux of the marketability of PNDs and LBS lies in the utility of navigation services to users. Most existing navigation services, however, do not provide multiple paths so that users can choose the path that best fits their needs. Instead, existing services provide the user with a path, referred to as the 'optimal path,' which is based on a proprietary, hidden algorithm (Park *et al*., 2002; Mohamed and Abdalla, 2004; Jeong *et al*., 2007). For example, existing navigation services provide only an 'optimal' path, even though the travel time difference between the 'optimal' path and the 'alternative' path is only one minute. On the other hand, drivers may prefer some other attributes, such as familiarity, driving convenience, and fewer right/left turns, over saving just one minute of driving time. The necessity of providing the multiple paths has been also presented by some literature such as Lee (1994), Park *et al*. (1997), Lim and Kim (2005) and Jeong *et al*. (2007).

Navigation services should focus on satisfying the needs of current customers as well as ways to attract new customers. Therefore, they should provide a variety of information that their users wish to have. The K shortest paths algorithm, which provides K multiple paths, has been considered to be a suitable algorithm for navigation services to use. However, it has the limitation of frequently providing alternative paths that are similar to paths already found in previous searches. The development of an algorithm to search alternative paths that correspond to user-specified conditions is expected to contribute to satisfying the various needs of current and potential users. Furthermore, the navigation services should be able to use efficient, path-finding algorithms to quickly provide alternative paths until the users choose a preferred alternative path.

The purpose of this paper is to present the development of an algorithm that provides dissimilar, alternative paths for a given origin and destination and to demonstrate the effectiveness of the algorithm by applying it to real networks. The alternative paths provided by the developed algorithm satisfy the user-specified

*Ph.D. Candidate, Dept. of Civil and Environmental Engineering, Seoul National University, Seoul 152-744, Korea (E-mail: iskra3138@gmail.com)
**Endowed Professor, Dept. of Urban and Regional Planning, University of Illinois at Urbana-Champaign, Champaign, IL 61820, USA (E-mail: tjohnkim@uiuc.edu)
***Member, Professor, Dept. of Civil and Environmental Engineering, Seoul National University, Seoul 152-744, Korea (E-mail: parkch@snu.ac.kr)
****Member, BK21 Assistant Professor, Dept. of Civil and Environmental Engineering, Seoul National University, Seoul 152-744, Korea (Corresponding Author, E-mail: kimdk95@snu.ac.kr)

allowable shared length ratio with already searched paths and travel cost ratio against the shortest path. In this paper, empirical studies based on real data are presented to evaluate the efficiency of the developed algorithm. The results show that the algorithm can provide dissimilar alternative paths efficiently by executing the shortest-path algorithm only once.

The remainder of this paper is organized as follows. The next section reviews existing path-finding algorithms, focusing on searching multiple alternatives. Then, the following section examines a new, efficient, dissimilar, alternative-path search algorithm for navigation services. After that, the new algorithm is illustrated using a toy network and tested on a Chicago network including the sensitivity analysis. Last, a summary and discussion are presented.

## 2. Review of Existing Pertinent Algorithms

The K shortest-paths algorithm, one of several existing algorithms, searches paths in the order of costs and provides K multiple paths based on the costs between an origin and a destination. In general, there are two methods for searching K numbers of multiple paths (Park, 1998) i.e., the label setting/label correcting method (Shier, 1976; 1979) and the path deletion method (Yen, 1971; Martins, 1984; Azevedo *et al*., 1993). The former method has to search all over again if a user rejects the paths provided. The latter method can search further for the $(K+1)^{th}$ path sequentially, based on the previously searched paths, even if the user rejects all K paths provided. The latter method is considered to be appropriate for the solution of multi-route problems in navigation services due to its effectiveness in continuing to provide alternative paths until the user makes a choice.

A set of multiple paths thus found by the classic K shortest-path algorithm includes very similar paths between an origin and a destination, oftentimes differentiated by one different link, while all other links in the paths are the same. Some of these paths are unreasonable and cannot satisfy a variety of individual needs. In order to avoid unreasonable searches for the $K^{th}$ best path, the algorithm should be able to avoid duplicate links from the path chosen in the $k^{th}$ search (k=1, … , K-1). To do this, Barra *et al*. (1993) proposed a link penalty method and Lim and Kim (2005) proposed the algorithm capable of considering path overlap using a link penalty.

Park *et al*. (2002) suggested an exact approach to identify multiple reasonable alternative paths considering not only overlaps but also costs in transportation networks. This method is based on the efficient vector labeling approach. A network is pruned using a resource (travel time) constraint. In addition, a path constraint is applied to maintain the uniqueness of alternative paths. A dominance-checking technique is introduced to avoid subpaths for each node that cannot satisfy travel time or path uniqueness requirements. This method, however, is not suitable for the condition which the travel time constraint is weak such as social, shopping, and recreational trips since this method prunes the search space on the basis of the time constraint.

Furthermore, Park *et al*. (2002) only selected the minimum cost path among those that satisfied the acceptable duplication rate as the next alternative path. However, users may want to acquire another path that reflects their preferences, such as sightseeing, safety, familiarity, and driving convenience, rather than minimum travel cost. Navigation services are required to offer a variety of opportunities for their users to choose the alternative paths corresponding with their needs. They should, therefore, continue to select the least shared length path among those that satisfied the allowable travel cost as the next alternative path.

Jeong *et al*. (2007) developed a heuristic algorithm to select K paths including a specific activity out of the candidate set of alternatives to consider users' various needs. Jeong *et al*. (2007) were able to establish a candidate set quickly by executing the shortest-path algorithm only once. As in Park *et al*. (2002), they, however, did not consider the duplication rates among the alternative paths. The search algorithm for navigation services should be able to provide alternatives that satisfy users' needs efficiently. Therefore, the algorithm developed in this paper selects the alternative path out of candidate alternatives based on Jeong *et al*. (2007) and includes user-specified allowable travel cost and duplication ratios.

## 3. Development of an Efficient Solution Algorithm

A shortest-path tree is assumed to exist from all origins to a destination, as described by Dreyfus (1969). In order to find the shortest paths from all node $i$ to a node $s$ when $k$=1 ($P_1^{is}$), a reverse Dijkstra algorithm[1)] was employed. (See Ahuja, Magnanti, and Orlin, 1993, for details.) Because the travel cost for each path is stored, the algorithm developed in this paper searches for the alternative paths by executing the shortest-path algorithm only once. In order to increase solution efficiency, Jeong *et al*. (2007) introduced a set of new variables, $C^{rj}$. If $R_{k+1}^{rj}$ is included in $C^{rj}$, we do not need to construct $R_{k+1}^{rj}$, $R_{k+1}^{r(j-1)}$, ..., $R_{k+1}^{r(r+1)}$, $R_{k+1}^{rr}$ since they have been searched and processed in the previous steps. As in Park *et al*. (2002), we can set the user-specified allowable shared length ratio $y$ for searching $P_k^{rs}$. Likewise, we can also set the user-specified allowable travel cost ratio $x$ for searching $P_k^{rs}$. The appropriate path satisfying the $x$ and $y$ conditions in a set of candidates becomes the $k^{th}$ alternative path from an origin $r$ to a destination $s$, $P_k^{rs}$. In this paper, we explain the choice of the least shared length path for the appropriate $k^{th}$ alternative path. Detailed steps in implementing the proposed algorithm are as follows:

Step 1. Finding $P_1^{is}$

a) Find all-to-one shortest paths from each node in the network to a destination $s$, using a reverse Dijkstra algorithm. Among those paths, the shortest path from a node $i$ to a destination $s$ is

1) A reverse Dijkstra algorithm determines the shortest paths from every node to destination node, while the Dijkstra algorithm determines the shortest paths from an origin node to every node.

set as $P_1^{is}$, and the travel cost for each path is $C[P_1^{is}]$. Then, the shortest path is $P_1^{rs}$.

Step 2. Finding $P_2^{rs}$, ..., $P_k^{rs}$, ..., $P_K^{rs}$

In order to find $P_K^{rs}$, the shortest paths $P_1^{rs}$, $P_2^{rs}$, ..., $P_{K-1}^{rs}$ must have been previously determined.

Sub-step 2-1. Construct $R_K^{rj}$.

a) Using $P_{K-1}^{rs}$, construct a sub-path, $R_K^{rj}$, by deleting links one by one from the nearest node from a destination $s$ to the furthest node in the previous shortest path. For example, if $P_{K-1}^{rs}$ is <($r$), (2), (3), (4), ($s$)>, then $R_K^{r4}$=<($r$), (2), (3), (4)>, $R_K^{r3}$=<($r$), (2), (3)>, $R_K^{r2}$=<($r$), (2)>, and $R_K^{rr}$=<($r$)>.

b) If $R_K^{rj}$ is <($r$), (2), ..., ($j$-1), ($j$)> and ($R_K^{rj} \in C^{rj}$), we do not need to consider $R_K^{rj}$, $R_K^{rj-1}$, ..., $R_K^{r2}$, $R_K^{rr}$ since they have been searched and processed in the previous steps.

c) If $R_K^{rj} \notin C^{rj}$, $C^{rj} = C^{rj} \cup \{R_k^{rj}\}$.

Sub-step 2-2. Update $B^{rs}$.

The shortest path to a destination $s$ from a node $h$ that is connected to a node $j$ in $R_K^{rj}$, but is not included in $P_{K-1}^{rs}$, was denoted as $P_1^{hs}$. The path, $R_K^{rj}$, plus $P_1^{hs}$ is added to $B^{rs}$. For example, if $P_{K-1}^{rs}$ is <($r$), (2), (3), ($s$)>, $R_K^{r3}$=<($r$), (2), (3)>, $P_1^{4s}$=<(4), (6), ($s$)>, $P_1^{5s}$=<(5), (7), (8), ($s$)>, and node (4), (5), (s) are connected to (3), then $R_K^{r3} + P_1^{4s}$=<($r$), (2), (3), (4), (6), ($s$)> and $R_K^{r3} + P_1^{5s}$=<($r$), (2), (3), (5), (7), (8), ($s$)> are added to $B^{rs}$.

Sub-step 2-3. Find $K^{th}$ path: $P_K^{rs}$.

If $P_c^{rs}$ is a path in the candidate set, the travel cost ratio of $P_c^{rs}$ is calculated as the ratio of $C[P_c^{rs}]$ to $C[P_1^{rs}]$, and the shared length ratios of $P_c^{rs}$ are calculated as the ratios of $L[P_k^{rs}, P_c^{rs}]$ to $L[P_k^{rs}]$ ($k$=1, ..., $K$-1).

a) If there are paths in $B^{rs}$ that meet the $x$ and $y$ conditions, the least shared length path becomes $P_K^{rs}$. $P_K^{rs}$ is added to $A^{rs}$ and removed from $B^{rs}$.

$$A^{rs} = A^{rs} \cup \{P_K^{rs}\},\ B^{rs} = B^{rs} - \{P_K^{rs}\}$$

b) If there are no paths in $B^{rs}$ that meet the $x$ and $y$ conditions, the least shared length path $P^{rs}$ is selected for next step 2 and is removed from $B^{rs}$.

$$B^{rs} = B^{rs} - \{P^{rs}\}.$$

The workings of the proposed algorithm stated above are shown in Fig. 1.

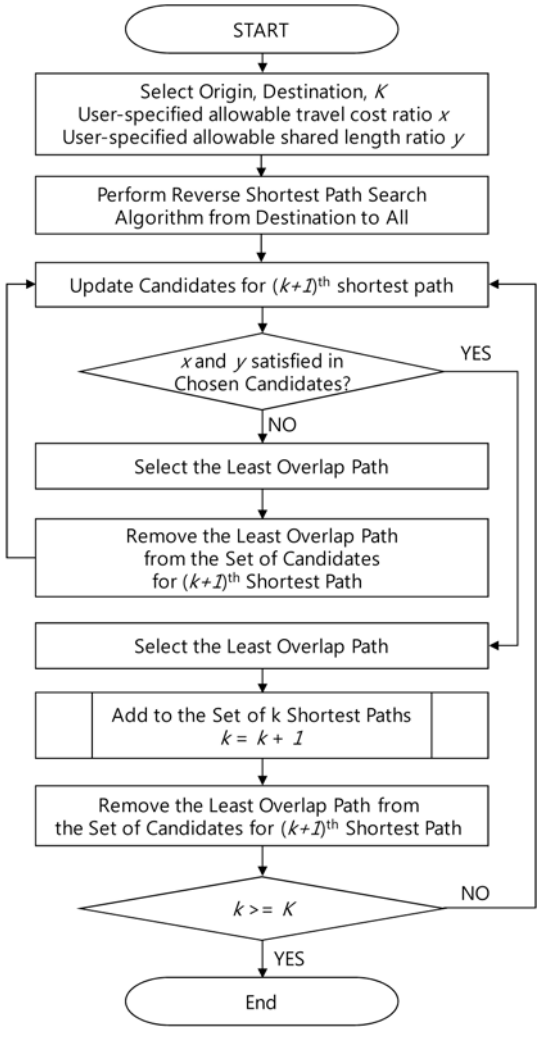


Fig. 1. Flow Chart for the Proposed K Shortest-Paths Algorithm for Navigation Services

## 4. Implementation of the Proposed Algorithm

### 4.1 Application to a Toy Network

The proposed solution algorithm was implemented using a toy network, as shown in Fig. 2. The toy network consists of 7 nodes and 12 links. In Fig. 2, the alphabetical characters within each circle denote node indices, and the numbers in front of the parentheses indicate link indices. The two numbers within parentheses designate costs and lengths of each link, respectively. In the example, the user-specified allowable shared length ratio is set to 0.7, and the user-specified allowable travel cost ratio is set to 1.7. The purpose of the application of the algorithm for the toy network is to demonstrate how the algorithm works.

In Step 1, using a reverse Dijkstra algorithm, the shortest paths to the destination $s$ when $k$=1, ($P_1^{is}$) are found, as shown in Fig.

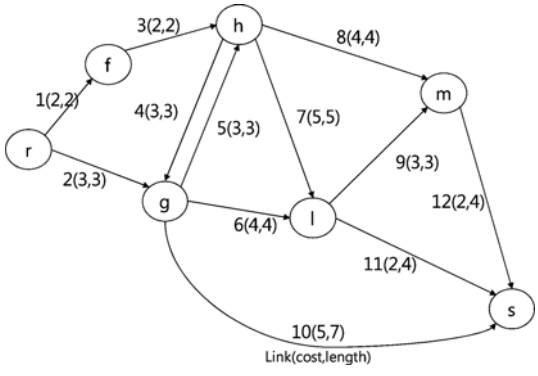


Fig. 2. Toy Network

3. Table 1 shows the summary information of $P_1^{is}$.

In Step 2, the shortest path between nodes $r$ and $s$ when $k$=2 ($P_2^{rs}$) is found based on the previously obtained $P_1^{rs}$ =<($r$), ($g$), ($s$)>. First, delete link 10 between nodes $g$ and $s$. The resulting sub-path $R_2^{rg}$ =<($r$), ($g$)> will then be stored in $C^{rg}$. Link 5 connecting nodes $g$ and $h$ and link 6 that connects nodes $g$ and $l$ will be the candidates for the next path search. The revised path cost is estimated using information stored in $R_2^{rg}$ and $P_1^{hs}$ =<($h$), ($m$), ($s$)>, i.e., 3 units from $R_2^{rg}$, 3 units from link 5 that connects node $g$ and $h$, and 6 units from $P_1^{hs}$ for the path <($r$), ($g$), ($h$), ($m$), ($s$)>. The sub-path cost between $r$ and $g$ in $R_2^{rg}$ can be found by subtracting the cost (5) of the deleted link 10 from the total cost (8) found between $r$ and $s$ in $P_1^{rs}$. Likewise, unit 3 from $R_2^{rg}$ and unit 4 for link 6 are added to the cost (2) from $P_1^{ls}$ for the path <($r$), ($g$), ($l$), ($s$)>. Next, link 2 is deleted from the network, and the next search for paths continues by repeating the process. The results are shown in Table 2. By the same process, the results are given in Table 3 when $k$=3 ($P_3^{rs}$). In these two tables, CR and LR denote the travel cost ratio and the shared length ratio, respectively.

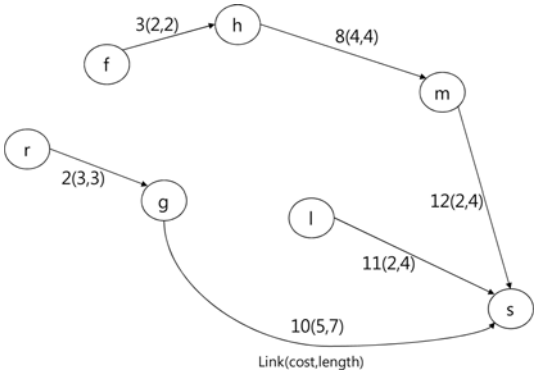


Fig. 3. Shortest Paths to Destination $s$

Table 1. Shortest Paths from All Nodes to the Destination $s$ when $k$=1 ($P_1^{is}$)

| Origin ($i$) | $r$ | $f$ | $g$ | $h$ | $l$ | $m$ |
|---|---|---|---|---|---|---|
| Cost | 8 | 8 | 5 | 6 | 2 | 2 |
| length | 10 | 10 | 7 | 8 | 4 | 4 |

## 4.2 Application to a Chicago Network

### 4.2.1 Data

The algorithm developed in this paper was also applied to a real Chicago network[2]. The network consisted of 12,982 nodes and 39,018 links. A part of the network is shown in Fig. 4. We

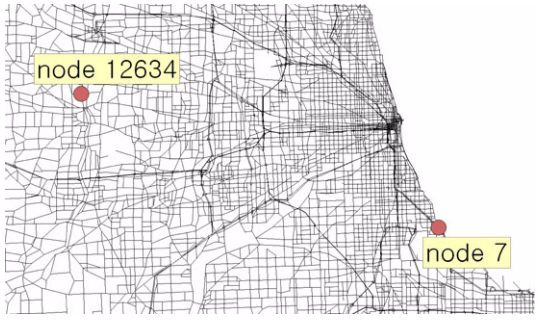


Fig. 4. A Section of the Chicago Network

Table 2. Results from the Application to the Toy Network ($k$=2)

| $K$ | $C^{rj}$[a] | $B^{rs}$ | $C[R_k^{ri}]+c^{ij}+C[P_1^{js}]$ | $CR$ | $LR$ | $A^{rs}$ | $C[P^{rs}]$ |
|---|---|---|---|---|---|---|---|
| 1 | | | | 1 | - | $r$, $g$, $s$ | 8 |
| 2 | $r$, $g$<br>$r$, $g$<br>$r$ | $r$, $g$, $h$, $m$, $s$<br>$r$, $g$, $l$, $s$<br>$r$, $f$, $h$, $m$, $s$ | (8-5)+3+6=12<br>3+4+2=9<br>(3-3)+2+8=10 | 1.5<br>1.13<br>1.25 | 0.3<br>0.3<br>0 | $r$, $f$, $h$, $m$, $s$ | 10 |

[a] $j$ denotes the last node in the sub-path.

Table 3. Results from the Application to the Toy Network ($k$=3)

| $K$ | $C^{rj}$ | $B^{rs}$ | $C[R_k^{ri}]+c^{ij}+C[P_1^{js}]$ | $CR$ | $LR$ | | $A^{rs}$ | $C[P^{rs}]$ |
|---|---|---|---|---|---|---|---|---|
| 1 | | | | 1 | - | | $r$, $g$, $s$ | 8 |
| 2 | $r$, $g$<br>$r$, $g$<br>$r$ | $r$, $g$, $h$, $m$, $s$<br>$r$, $g$, $l$, $s$<br>$r$, $f$, $h$, $m$, $s$ | (8-5)+3+6=12<br>3+4+2=9<br>(3-3)+2+8=10 | 1.5<br>1.13<br>1.25 | 0.3<br>0.3<br>- | 0.67<br>0<br>- | $r$, $f$, $h$, $m$, $s$ | 10 |
| 3 | $r$, $f$, $h$, $m$<br>$r$, $f$, $h$<br>$r$, $f$, $h$<br>$r$, $f$<br>$r$ | $r$, $f$, $h$, $m$ $(X)$[a]<br>$r$, $f$, $h$, $g$, $s$<br>$r$, $f$, $h$, $l$, $s$<br>$r$, $f$ $(X)$<br>$r$ $(X)$[b] | (10-2)<br>(8-4)+3+5=12<br>4+5+2=11<br>-<br>- | -<br>1.75[c]<br>1.63<br>-<br>- | -<br>-<br>0<br>-<br>- | -<br>-<br>0.33<br>-<br>- | $r$, $g$, $l$, $s$ | 9 |

[a] No other link is connected to the node, so the search stops.
[b] Due to existence of the previously considered sub-path in $C^{rj}$, the search stops.
[c] The calculation of the shared length ratio stops because the cost ratio is bigger than the maximum allowed travel cost ratio 1.7.

2) These data were downloaded from http://www.bgu.ac.il/~bargera/tntp/.

Table 4. Sample Link Data from the Chicago Network

| Link No. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ··· |
|---|---|---|---|---|---|---|---|---|---|---|
| From Node | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ··· |
| To Node | 10293 | 10294 | 10295 | 10296 | 10297 | 10298 | 10299 | 10300 | 10301 | ··· |
| Cost | 0.1125 | 0.125 | 0.0975 | 0.145 | 0.125 | 0.145 | 0.1125 | 0.125 | 0.11 | ··· |
| Length | 251.048 | 251.048 | 251.048 | 251.048 | 251.048 | 251.048 | 251.048 | 251.048 | 251.048 | |

Table 5. Sample Node Location Data from the Chicago Network

| Node No. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ··· |
|---|---|---|---|---|---|---|---|---|---|---|
| X Coordinate | 712,475 | 713,275 | 715,275 | 713,675 | 717,675 | 713,775 | 717,975 | 716,775 | 718,175 | ··· |
| Y Coordinate | 1,855,780 | 1,850,580 | 1,847,980 | 1,844,580 | 1,842,680 | 1,839,180 | 1,838,980 | 1,831,380 | 1,835,380 | ··· |

used a cost in the 'ChicagoRegional_flow.txt' for link cost, and we calculated Euclidean distance with the coordinates X and Y in the 'ChicagoRegional_node.txt' for link length.

The sample data are shown in Tables 4 and 5. The highest link cost is 36.9967902 from node 11933 to node 9260. The lowest link cost is 0.025 from node 2063 to node 2062. The longest link length is 88,796.8 from node 1782 to node 7614. The shortest link length is 80.7257 from node 8295 to node 8622. Node locations expand between 349,600 and 840,669 in the *x* coordinates and between 1,572,888 and 2,193,600 in the *y* coordinates. We used $C^{++}$ for programming the computer and implemented the solution algorithm using a computer with Pentium Core2 duo 1.86 GHz processor, a Windows XP operating system, and 2 GB DDR RAM.

### 4.2.2 Selection of the Least Shared Length Path for the $k^{th}$ Alternative Path

From node 12634 to node 7, we obtained the 'optimal' path and three dissimilar alternative paths using the proposed algorithm to choose the least shared length path for the $k^{th}$ alternative path, as shown in Figs. 5, 6, 7 and 8 and in Tables 6 and 7.

Table 6 shows the basic information of the paths, and Table 7 shows the shared length ratios between paths. As shown in Table 8, It took 0.266 second to find the four paths when the user-specified allowable travel cost ratio was set to 1.05 (5%) and the user-specified allowable shared length ratio was set to 0.7 (70%). Alternative path 1 requires 4.3% more travel costs than the shortest path, while the shared length ratio with the shortest path is only about 45%. The travel cost of alternative path 2 is 1.56% larger than the cost of the shortest path, and the shared length ratios for the shortest path and alternative path 1 are 65% and

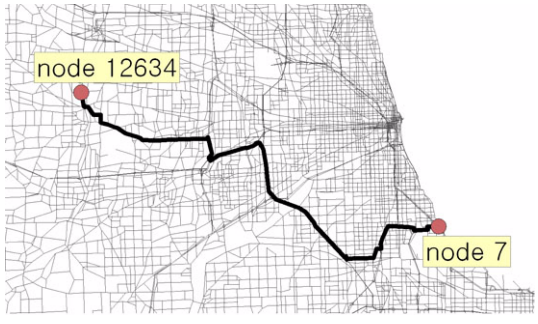


Fig. 5. Shortest Path

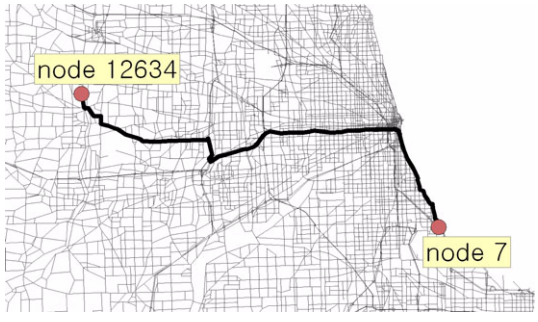


Fig. 6. Alternative Path 1 (When we choose the least shared length path.)

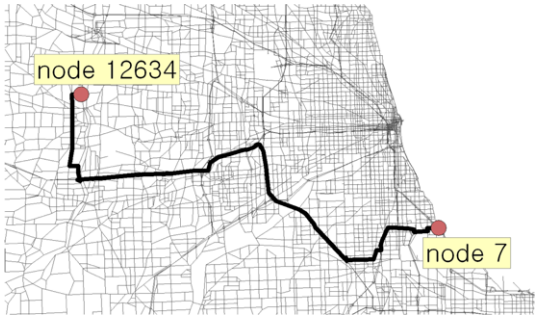


Fig. 7. Alternative Path 2 (When we choose the least shared length path.)

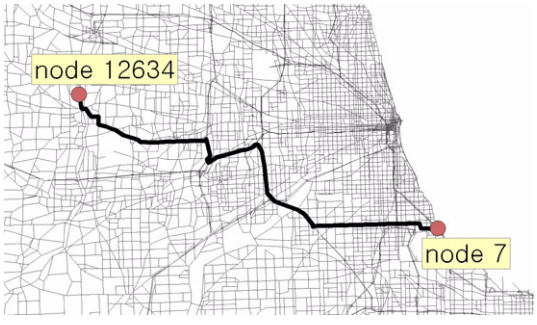


Fig. 8. Alternative Path 3 (When we choose the least shared length path.)

Table 6. Test Results (When we choose the least shared length path.)

| | Shortest Path | Alternative Path 1 | Alternative Path 2 | Alternative Path 3 |
|---|---|---|---|---|
| Number of Links | 87 | 109 | 80 | 89 |
| Cost | 114.08 | 119.039 | 115.864 | 117.307 |
| Length | 310,861 | 295,776 | 350,651 | 283,218 |

Table 7. Shared Length Ratios (When we choose the least shared length path.)

| | Shortest path | Alternative Path 1 | Alternative Path 2 | Alternative Path 3 |
|---|---|---|---|---|
| Shortest path | 1 | 0.448288 | 0.649725 | 0.681568 |
| Alternative Path 1 | 0.448288 | 1 | 0.103012 | 0.471152 |
| Alternative Path 2 | 0.649725 | 0.103012 | 1 | 0.2937 |
| Alternative Path 3 | 0.681568 | 0.471152 | 0.2937 | 1 |

10%, respectively. The travel cost for alternative path 3 is 2.83% larger than the cost for the shortest path, while the shared length ratios with the shortest path, with the alternative path 1, and with alternative path 2 are 68%, 47%, and 29%, respectively. In the test, the average cost ratio is 2.9% and the average shared length ratio is 44%.

#### 4.2.3 Selection of the Minimum Cost Path for the $k^{th}$ Alternative Path

Next, we obtained the 'optimal' path and three dissimilar alternative paths using the proposed algorithm to choose the minimum cost path for the $k^{th}$ alternative path for the same conditions. As shown in Figs. 9, 10 and 11 and in Tables 8, 9 and 10, it took 0.250 second to find the four paths.

Table 9 shows the basic information for the paths, and Table 10 shows the shared length ratios between paths. Alternative path 1 requires 1.6% more travel costs than the shortest path, while the shared length ratio with the shortest path is only about 65%. The travel cost of alternative path 2 is 2.83% larger than the travel cost of the shortest path, and the shared length ratios with the shortest path and with alternative path 1 are, 68% and 29%, respectively. The travel cost for alternative path 3 is 4.35% larger than the travel cost for the shortest path, while the shared length ratios with the shortest path, with the alternative path 1, and with alternative path 2 are, 45%, 9%, and 49%, respectively. In the

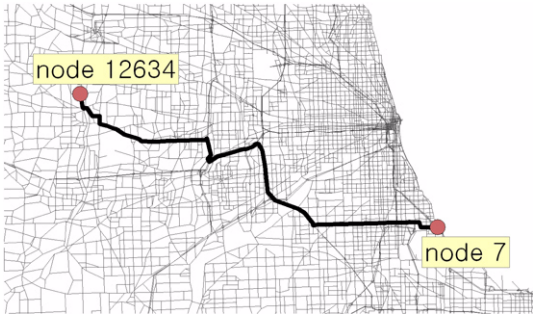


Fig. 10. Alternative Path 2 (When we choose the minimum cost path.)

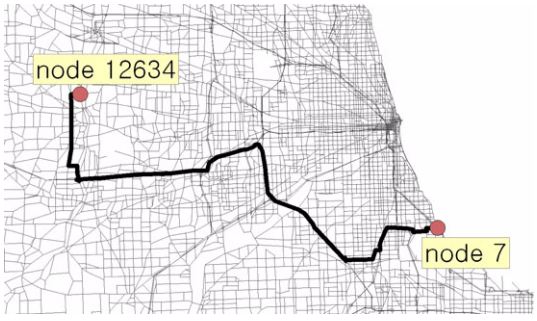


Fig. 9. Alternative Path 1 (When we choose the minimum cost path.)

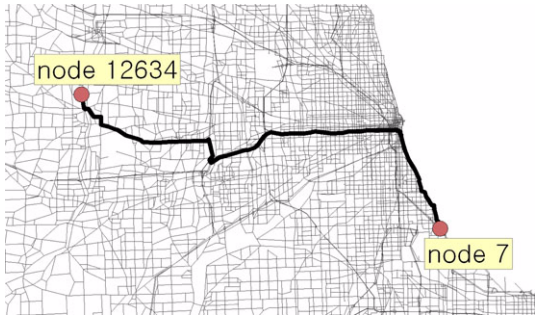


Fig. 11. Alternative Path 3 (When we choose the minimum cost path.)

Table 8. Computation Time

| | The computation time (second) | | |
|---|---|---|---|
| | Total | Step 1 (all-to-one) | Step 2 (3 alt. paths) |
| Selection of the least shared length path | 0.266 | 0.156 | 0.110 |
| Selection of the minimum cost path | 0.250 | 0.157 | 0.093 |

test, the average cost ratio is 2.9% and the average shared length ratio is 44%.

#### 4.2.4 Sensitivity Analysis

Lastly, we performed the sensitivity analysis of the proposed algorithm with changing the user-specified allowable travel cost ratio $x$ and the user-specified allowable shared length ratio $y$. After selecting randomly 1,000 O-D pairs, we obtained the 'optimal' path and three dissimilar alternative paths using the proposed algorithm to choose the least shared length path for the $k^{th}$ alternative path, as shown in Tables 11 and 12. The path search process for each O-D pair was performed for 3 second, and the cases where computation time is more than 3 second were considered as the Failure cases. Table 11 shows the number

Table 9. Test Results (When we choose the minimum cost path.)

| | Shortest Path | Alternative Path 1 | Alternative Path 2 | Alternative Path 3 |
|---|---|---|---|---|
| Number of Links | 87 | 80 | 89 | 109 |
| Cost | 114.08 | 115.864 | 117.307 | 119.039 |
| Length | 310,861 | 350,651 | 283,218 | 295,776 |

Table 10. Shared Length Ratios (When we choose the minimum cost path.)

| | Shortest path | Alternative Path 1 | Alternative Path 2 | Alternative Path 3 |
|---|---|---|---|---|
| Shortest path | 1 | 0.649725 | 0.681568 | 0.448288 |
| Alternative Path 1 | 0.649725 | 1 | 0.2937 | 0.0868912 |
| Alternative Path 2 | 0.681568 | 0.2937 | 1 | 0.492042 |
| Alternative Path 3 | 0.448288 | 0.0868912 | 0.4942042 | 1 |

Table 11. Sensitivity Analysis

| $x$ | $y$ | SUCCESS | | | | | Failure |
|---|---|---|---|---|---|---|---|
| | | Total | $K$=0 | $K$=1 | $K$=2 | $K$=3 | |
| 1.05 | 0.7 | 642 | 173 | 84 | 36 | 349 | 358 |
| 1.05 | 0.8 | 815 | 112 | 83 | 46 | 574 | 185 |
| 1.05 | 0.9 | 969 | 78 | 38 | 31 | 822 | 31 |
| 1.10 | 0.7 | 645 | 79 | 42 | 28 | 496 | 355 |
| 1.10 | 0.8 | 828 | 51 | 28 | 28 | 721 | 172 |
| 1.10 | 0.9 | 981 | 36 | 19 | 11 | 915 | 19 |
| 1.20 | 0.7 | 681 | 35 | 23 | 12 | 611 | 319 |
| 1.20 | 0.8 | 870 | 22 | 16 | 12 | 820 | 130 |
| 1.20 | 0.9 | 987 | 15 | 9 | 8 | 955 | 13 |

Table 12. Average Time, Cost Ratio and Shared Length Ratio

| $x$ | $y$ | $K$=0 | $K$=1 | | | $K$=2 | | | $K$=3 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | time | time | cost | shared length | time | cost | shared length | time | cost | shared length |
| 1.05 | 0.7 | 0.31 | 0.48 | 1.03 | 0.58 | 0.55 | 1.03 | 0.54 | 0.48 | 1.03 | 0.46 |
| 1.05 | 0.8 | 0.23 | 0.46 | 1.03 | 0.69 | 0.54 | 1.03 | 0.66 | 0.34 | 1.03 | 0.56 |
| 1.05 | 0.9 | 0.17 | 0.34 | 1.03 | 0.77 | 0.24 | 1.03 | 0.72 | 0.23 | 1.02 | 0.64 |
| 1.10 | 0.7 | 0.36 | 0.46 | 1.06 | 0.59 | 0.42 | 1.06 | 0.51 | 0.39 | 1.05 | 0.45 |
| 1.10 | 0.8 | 0.21 | 0.66 | 1.07 | 0.73 | 0.27 | 1.05 | 0.63 | 0.29 | 1.05 | 0.53 |
| 1.10 | 0.9 | 0.19 | 0.33 | 1.06 | 0.79 | 0.18 | 1.04 | 0.69 | 0.22 | 1.04 | 0.60 |
| 1.20 | 0.7 | 0.22 | 0.23 | 1.11 | 0.59 | 0.26 | 1.12 | 0.53 | 0.40 | 1.07 | 0.45 |
| 1.20 | 0.8 | 0.24 | 0.19 | 1.14 | 0.65 | 0.17 | 1.11 | 0.64 | 0.29 | 1.07 | 0.53 |
| 1.20 | 0.9 | 0.15 | 0.16 | 1.15 | 0.64 | 0.41 | 1.09 | 0.76 | 0.21 | 1.06 | 0.59 |

of Success and Failure cases, where K is the number of the searched alternative paths for 3 second. Table 12 shows the average time, cost ratio and shared length ratio for the Success cases.

Based on the results above, the algorithm developed in this paper can search 3 dissimilar alternative paths, with travel costs that are similar to the cost of the shortest path, within reasonable search times.

## 5. Conclusions

In this paper, we suggest a heuristic algorithm that searches dissimilar alternative paths efficiently for navigation services. Navigation service users prefer to have multiple options in order to select the most appropriate path considering all available information. Generation and provision of alternative choices would offer more flexible alternatives for users to choose from, particularly in cases in which there is a difference of only a few minutes between the "best" path and some alternative paths. Path search algorithms for navigation services can also find appropriate multiple alternative paths quickly, based on the user's needs. To do this, we made a set of candidate paths with the method proposed by Jeong *et al*. (2007) and selected an appropriate path for the k$^{th}$ alternative path in the set.

We demonstrated how the algorithm works using both a toy network and a Chicago network. We established the candidate set that satisfied the user-specified allowable travel cost ratio and shared length ratio at the same time. Then, among the available paths, we selected the least shared path and the minimum cost path as the alternative path, respectively. The shortest path and three alternative paths were searched quickly enough for convenient use by navigation services. Users can obtain alternative paths according to their preferences by controlling the user-specified ratios.

The solution algorithm developed in this paper will also be useful for location-based concierge services (Kang and Kim, 2005; Kang *et al*., 2006; Kim, 2004; Jeong *et al*., 2007) where a number of activities must be accomplished to select the preferred path. A user may wish to stop by a bakery, a flower shop, and a card shop on the way home from work. In so doing, he/she would like to select the preferred path if multiple routes are presented, which have associated costs that are slightly different from the cost of the 'optimal' path. The need for informing users of the alternative routes will become a critical factor for marketing LBS, since customized services can be provided for different users according to the users' profiles.

## Notations[3)]

$\mathbf{A}^{rs}$: Set of k shortest paths between an origin $r$ ($r \in \mathbf{O}$) and a destination $s$ ($s \in \mathbf{D}$)

$a$, $b$: Index for links

$\mathbf{B}^{rs}$: Set of candidates for the (k+1)$^{th}$ dissimilar alternative path between an origin $r$ and a destination $s$

$C[P_k^{rs}]$ : Travel cost of a path $P_k^{rs}$

$\mathbf{C}^{rj}$ : Set of $R_1^{rj}$ , ..., $R_k^{rj}$

$c_a^{ij}$ : Travel cost of a link $a$ that connects nodes $i$ and $j$

$i$, $j$: Index for nodes

$k$: The number of alternative paths, $k$=1, 2, ..., $K$

$L[P_k^{rs}]$ : Travel length of a path $P_k^{rs}$

$L[P_k^{rs}, P_c^{rs}]$ : Shared length of path $P_k^{rs}$ and $P_c^{rs}$

$\mathbf{O}$, $\mathbf{D}$: Set of origin nodes and destination nodes, respectively

$P_c^{rs}$ : The candidate path from an origin $r$ to a destination $s$ ($P_c^{rs} \in \mathbf{B}^{rs}$), which is denoted as a sequence of traversed nodes ($P_c^{rs}$ =<($r$), (1), (2), ..., ($s$) >)

$P_k^{rs}$ : The k$^{th}$ dissimilar alternative path from an origin $r$ to a destination $s$ ($P_k^{rs} \in \mathbf{A}^{rs}$), which is denoted as a sequence of traversed nodes ($P_k^{rs}$ =<($r$), (1), (2), ..., ($s$) >)

$R_k^{rj}$ : The sub-path of $P_{k-1}^{rs}$ from an origin $r$ to a node $j$ ($R_k^{rj}$ =<($r$), (1), (2), ..., ($j$) >)

$x$: User-specified allowable travel cost ratio against the shortest path

$y$: User-specified allowable shared length ratio with already searched paths

3) Superscript denotes nodes and subscript denotes links or the k$^{th}$ search.

## Acknowledgements

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2006-511-D00084).

## References

Ahuza, R. K., Magnanti, T. L., and Orlin, J. B. (1993). *Network flows: Theory, alogorithm, and applications*, Prentice Hall, N.J.

Azevedo, J. A., Costa, M. E. O. S., Madeira, J. J. E. R. S., and Martins, E. Q. V. (1993). "An algorithm from the ranking of shortest paths." *European Journal of Operational Research*, Vol. 69, No. 1, pp. 97-106.

Barra, T., Perez, B. and Anez, J. (1993). "Multidimensional path search and assignment." *Proc. 21st PTRC Summer Annual Conference*, PTRC, Manchester, England, pp. 307-319.

Berg Insight (2008). *Personal navigation devices*, Gothenburg, Sweden, http://www.berginsight.com/ShowReport.aspx?m_m=3&id=77, (Accessed Apr. 6, 2009).

Dreyfus, S. E. (1969). "An appraisal of some shortest-path algorithms." *Operations Research*, Vol. 17, No. 3, pp. 395-412.

Jeong, Y. J., Hong, Y., and Kim, T. J. (2007). "A flexible multi-path search algorithm for multi-purpose location-based activities." *Transportation Research Record*, No. 2039, pp. 50-57.

Kang, S. and Kim, T. J. (2005). "Solving a location-based concierge service problem: A heuristic approach using genetic algorithm." *Presented at the Symposium on Societies and Cities in the Age of Instant Access*, University of Utah, Salt Lake City, UT.

Kang, S., Oh, S., and Kim, T. J. (2006). "A heuristic algorithm for

solving a multimodal location-based concierge service problem." *Transportation Research Record*, No. 1972, pp. 123-132.

Kim, T. J. (2004). *Multi-modal routing and navigation cost functions for Location-Based Services (LBS)*, In Urban and Regional Transportation Modeling: Essays in Honor of David Boyce, D. H. Lee Ed., Edward Elgar, Northampton, M.A.

Lee, C. K. (1994). "Multiple-path Routing Strategy for Vehicle Route Guidance Systems." *Transportation Research Part C*, Vol. 2, No. 3, pp. 185-195.

Lim, Y. and Kim, H. (2005). "A shortest path algorithm for real road network based on path overlap." *Journal of the Eastern Asia Society for Transportation Studies*, Vol. 6, pp. 1426-1438.

Martins, E. Q. V. (1984). "An algorithm for ranking paths that may contain cycles." *European Journal of Operational Research*, Vol. 18, No. 1, pp. 123-130.

Mohamed, A. and Abdalla, M. F. (2004). "Modeling repeated multinomial route choices under advanced traveler information system: generalized estimating equations with polytomous logit function." *Transportation Research Record*, No. 1882, pp. 61-69.

Park, D. (1998). *Multiple path based vehicle routing in dynamic and stochastic transportation networks,* PhD Thesis, Department of Civil Engineering, Texas A&M University.

Park, D. and Rilett, L. R. (1997). "Identifying multiple and reasonable paths in transportation networks – A heuristic approach." *Transportation Research Record*, No 1607, pp. 31-37

Park, D., Sharma, S. L., Rilett, L. R., and Chang, M. (2002). "Identifying multiple reasonable alternative routes: Efficient vector labeling approach." *Transportation Research Record*, No. 1783, pp. 111-118.

Shier, R. D. (1976). "Iterative methods for determining the k shortest paths in a network." *Networks*, Vol. 6, No. 3, pp. 205-229.

Shier, R. D. (1979). "On algorithms from finding the k shortest paths in a network." *Networks*, Vol. 9, No. 3, pp. 195-214.

Yen, J. Y. (1971). "Finding the K shortest loopless paths in a network." *Management Science*, Vol. 17, No. 11, pp. 712-716.

http://www.lgeri.com/industry/electronic/listWebZine.asp?grouping=01030200&cPage=2&srchtype=1&srchword (Accessed Apr. 6, 2009).

http://www.bgu.ac.il/~bargera/tntp/ (Accessed Apr. 6, 2009).